AD A118154

(12)

Report MDA903-79-C-0190-02AE-1

FINAL TECHNICAL REPORT FOR LOW-COST PACKET RADIO DECODERS

W. ALLEN SHUMATE

M/A-COM LINKABIT, Inc.
10453 Roselle Street
San Diego, CA 92121

30 May 1982

Prepared for:

ADVANCED RESEARCH PROJECTS AGENCY
Attention: Dr. Robert Kahn,
Program Management
1400 Wilson Boulevard
Arlington, VA 22209

FINAL TECHNICAL REPORT

FOR

LOW-COST PACKET RADIO DECODERS

Contract MDA903-79-C-0190

ARPA Order Number: 3479

Effective Date: 20 February 1979

Expiration Date: 30 May 1982

W. Allen Shumate

M/A-COM LINKABIT, Inc.
10453 Roselle Street
San Diego, CA 92121

(714) 453-7007

30 May 1982

This research was sponsored by the
Defense Advanced Research Projects
Agency under ARPA Order No.: 3479
Contract No.: MDA903-79-C-0190
Monitored by Dr. Robert Kahn (IPTO)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: MDA903-79-C-0190-02AE-1	2. GOVT ACCESSION NO.: AD-A118 154
3. RECIPIENT'S CATALOG NUMBER: --	
4. TITLE (and Subtitle): FINAL TECHNICAL REPORT FOR LOW-COST PACKET RADIO DECODERS	5. TYPE OF REPORT & PERIOD COVERED: FINAL REPORT
	6. PERFORMING ORG. REPORT NUMBER: NONE
7. AUTHOR(s): W. ALLEN SHUMATE	8. CONTRACT OR GRANT NUMBER(s): MDA903-79-C-0190
9. PERFORMING ORGANIZATION NAME AND ADDRESS: M/A-COM LINKABIT, Inc. 10453 ROSELLE STREET SAN DIEGO CA 92121	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: NONE
11. CONTROLLING OFFICE NAME AND ADDRESS: ADVANCED RESEARCH PROJECTS AGENCY 1400 WILSON BOULEVARD ARLINGTON VA 22209	12. REPORT DATE: 30 MAY 1982
	13. NUMBER OF PAGES: --
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office): ---	15. SECURITY CLASS. (of this report): UNCLASSIFIED
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

APPROVED FOR PUBLIC RELEASE
DISTRIBUTION UNLIMITED

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

NONE

18. SUPPLEMENTARY NOTES

NONE

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

NONE

(large scale integrated)

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This is the final report for the development of an LSI Sequential Decoder Chip and an Interface Chip, the Direct Memory Access Interface, for use in the Low-Cost Packet Radio. The report describes each of the devices, the procedure used in development, and problems associated with each development. Additionaly, the functional specifications for each device are included. These specifications cover the internal architecture and applications for each device.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

Table of Contents

1. Packet Radio Sequential Decoder Final Report

1.1 Introduction

This document comprises the final report for the Low-Cost Packet Radio Decoder development, sponsored by DARPA under contract MDA903-79-0190. The report includes a brief program overview, the functional specification for the LS56 sequential decoder chip, and the functional specification for the Direct Memory Access Interface (DMA/I) chip. Both devices are the result of this development contract.

1.2 Program Overview

1.2.1 Decoder Development Results

The objective of this contract is the development of large scale integrated circuits to implement a sequential decoder system with a minimum number of support chips. The result of this development is two LSI devices: the LS56 sequential decoder chip and the DMA/I chip. The LS56 convolutionally encodes and sequentially decodes packet data or continuous data. The DMA/I interfaces the LS56 to a micro-processor data bus for sequential decoding in packet applications.

The LS56 sequential decoder encodes and decodes in rates 1/2, 3/4, and 7/8. The LS56 capabilities also include: continuous mode BPSK, QPSK, and offset QPSK interfaces; packet mode interfaces; and hard decision, soft decision, and erasure channel operation. The LS56 and DMA/I maximum computation rate is 1.5 MHz for the prototype devices. This computation rate allows data rates up to 450 KBPS in continuous operation and 200 KBPS in packet operation.

The LS56 is a fully custom integrated circuit, implemented in NMOS. The minimum feature size for the LS56 is five microns. The size of the LS56 die is 282 MIL X 242 MIL and it is packaged in a 68 pin LSI package. The device count for the LS56 is approximately 13,000 transistors.

The DMA/I chip is a semi-custom metal-gate CMOS gate array. The gate array is a standard array developed by California Devices, Inc. The metal layer of the gate array is the only portion of the chip that was developed for the DMA/I design. The array consists of 1000 gates, 800 of which are utilized in the DMA/I. The die size of the DMA/I is 225 MIL X 200 MIL and it is packaged in a 64-pin package.

Decoder systems based on the LS56 require some external support logic. In the packet mode, the decoder system includes one LS56, one DMA/I and 13 other MSI-TTL devices to implement a half-duplex encoder-decoder. The continuous mode application requires two LS56's and 20 MSI devices to implement a full-duplex encoder-decoder system.

1.2.2 Development Procedure

The LS56 development was a joint effort between LINKABIT and a subcontractor, Alphatron, Inc. LINKABIT'S responsibilities included system design, detailed logic design, and verification of the design using a TTL model. Alphatron converted LINAKBIT'S design into an NMOS compatible logic implementation, and generated a composite layout of the device. The composite layout was digitized, and masks were generated from this digitization. Prototype devices were fabricated from the masks. LINKABIT extensively tested these prototypes and found no functional logic

problems. Only one design-fabrication cycle from a functional logic design to working chips was required. This success is attributed to the extensive check steps performed at LINKABIT and Alphatron to insure a correct logic translation.

The DMA/I development incorporated the same procedure as the LS56 development. LINKABIT did the system design, detailed functional logic design and wrote a specification for the D.C. and A.C. parameters of the device. Additionally, LINKABIT verified the logic with another TTL model. The logic and specification was delivered to California Devices, Inc. (CDI), a company specializing in CMOS gate arrays. CDI did a logic conversion into a CMOS-compatible design and generated a composite layout for the metal layer of the DMA/I. Masks were generated and prototypes delivered to LINKABIT for functional testing. The DMA/I prototypes also worked on the first delivery of prototypes to all specifications.

1.2.3 Chip Development Problems

The development processes for both the LS56 and DMA/I each had several problems related to the time required to do the development and the number of devices required on each chip.

The original estimate for the LS56 device count was 4,000, but the final device count is approximately 13,000. Most of the problems in estimating device count stem from the additional features added to the chip and from a fundamental difference in the implementation of logic in MSI functions (TTL) versus LSI functions. The features added include interface logic required on-chip that would otherwise exist externally; additional I/O circuitry required in the NMOS implementation, for speed

considerations, that is not required in an MSI-TTL implementation; and problems with estimating device count without knowledge of the final logic design.

The DMA/I development encountered problems with selection of the gate array and gate array manufacturer. The first company selected as a vendor, Interdesign, was not supportive of the design effort. A second vendor, CDI, was then selected. Problems with CDI were related to delays in the digitization step because of equipment failure and operator inexperience. This stretched a 15-week development into a six month development.

1.3 Report Organization

The remainder of this report is organized into two sections:

Appendix I	The LS56 sequential decoder functional specification. This document includes a detailed description of the LS56 architecture along with descriptions of decoder systems based on the LS56.
Appendix II	The DMA/I chip functional specification. This document includes a description of the DMA/I architecture along with interfacing details.

I. LS56 LSI Sequential Decoder Functional Specification

FUNCTIONAL SPECIFICATION

LS56 LSI SEQUENTIAL DECODER

DRAFT

Revision H

28 May 1982

Table of Contents

List of Figures

List of Tables

1. Introduction

1.1 Decoder System Objectives

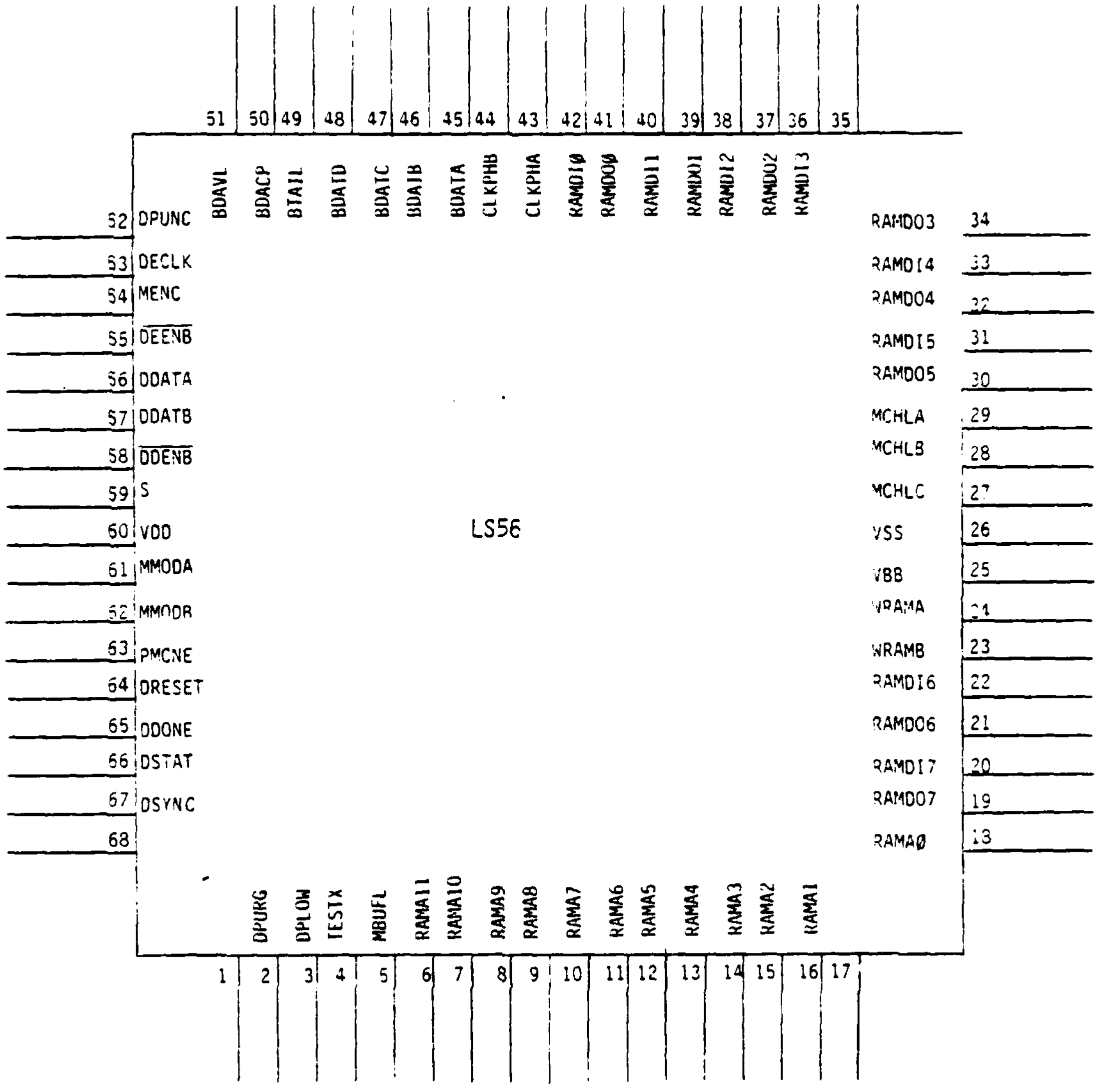
1.1.1 Specification Objectives

The objective of this specification is the definition of system, logic, timing and environment characteristics of a Large Scale Integrated (LSI) circuit denoted the LS56. This LSI device as specified is a component for use in digital communication systems requiring forward error correction (FEC) data processing. The LS56 is a digital-only device implemented with saturated logic techniques employing two clock phases. The LS56 by design may be incorporated into various communication systems : versatility of application is a key design objective. The logic symbol for the LS56 is given in Figure 1-1.

1.1.2 General Coding Objectives

The LS56 is a general-purpose encoder or decoder (simplex) which utilizes long convolutional codes in combination with a sequential decoding algorithm to provide enhanced Bit Error Rate (BER) performance across digital communications channels of several kinds. The specific coding objective is the reduction of required E_b/N_o (digital SNR) to acheive a particular output BER with a fixed modulation type. In order to accomplish this improvement in BER, the use of redundant codes is necessary. These codes generally trade off potential coding gain against increased channel bandwidth. In order to allow a choice of tradeoffs, the LS56 is a multiple code rate device, offering four effective coding rates: pseudo-1/4 plus true 1/2, 3/4 and 7/8. The particular coding gain realizable with the LS56 will depend

Figure 1-1. LS56 Logic Symbol



upon the type of channel utilized, the code rate selected, the decoding buffer memory size selected and the data rate. The latter two factors are direct results of the sequential decoding technique itself. As a definition, coding gain is equal to the reduction of E_b/N_o (in dB) possible in the coded system in order to match BER performance of the uncoded system.

The sequential decoding algorithm used in the LS56 is a derivative of the Fano algorithm. Some special provisions for block data operation have been added for efficiency in this mode. The Fano algorithm is a non-realtime searching procedure which proceeds according to the results of an arithmetic metric test in each consecutive search step. Generally, the ratio of required steps to bits decoded is greater than one (when noisy demodulator decisions are present) and the decoder is required to operate several times more rapidly than the uncoded data rate. Under typical conditions, there exists a queueing problem which may be described as the short-term average rate at which a decoding input buffer is served by the decoder versus a constant rate of arrival of newly received (encoded and noise-corrupted) channel bits. It is this queueing problem and its associated statistics which determines the performance of low-code rate sequential decoders. At high code rates (e.g. 7/8), the queueing problem is joined by an essentially independent code distance limitation which acts to reduce ideal coding gain. Ignoring the effects of limited code distance, a sequential decoder only contributes output errors (to raise the BER above zero) when the short-term buffer service rate falls critically below the new channel bit arrival rate. In this event, called buffer failure, the decoder is forced to abandon its non-realtime search and instead output undecoded channel bits which contain errors at a rate equal to

the demodulator hard decision (sign bit) error rate. Thus, the nature of output errors with sequential decoding is the appearance of occasional burst of errors. It is the probability of buffer failure around which complete decoding systems are built with the LS56.

The probability of buffer failure depends explicitly upon the available size of the decoder input buffer, the ratio of decoder computation rate to (uncoded) information rate and the channel error rate. Somewhat simplified, the probability of buffer failure, Pr[BF], under channel conditions which approach the limit of the code's inherent ability to detect (and allow correction of) errors may be expressed as:

$Pr[BF] = k\ (uB)^{-a}$ where k = a normalization constant
u = ratio of decoder/data rate
B = size of buffer memory
and $a = f(\text{channel error rate})$.

Since this probability pertains only to the onset of buffer failure at any given bit time, it is necessary to modify the expression to represent the average probability of output error from the decoder. This is done by noting the demodulator hard decision error rate, p, and the duration of a typical, isolated buffer failure which is strongly influenced by design considerations. In the LS56 in rate 7/8 for example, the duration is typically 5x168=840 output bits. Therefore:

$$Pr[\text{output error}] = BER = 840\ p\ k\ (uB)^{-a}.$$

This expression is evaluated for a particular p, k, u, B and "a" according to the system and channel in use. Although the approach above looks promising for calculating BER, it is noted

than the normalization constant k is not very amenable to analysis and that it is normally deduced in practice from measurements of real decoder performance against carefully controlled channel simulations. Lastly, in the LS56 the two expressions above require modification to account for the fact that input/output operations during extended searches steal machine cycles away from useful decoding; this effect is most pronounced at the highest data rates.

Coding acheives a reduction in required E_b/N_o for a given BER traded off against decoder complexity, increased bandwidth requirements and increased delay through the communication link. The LS56 approaches these tradeoffs with a variety of feature selections:

1. variable code rates: 1/4, 1/2. 3/4 and 7/8
2. variable buffer lengths B: either the set (128,256) or (1K, 4K)
3. selectable hard decision or 2-bit soft decision compatibility.

Additionally, the base design of the LS56 allows computation clock rates up to 1.5 MHz with future improved devices planned for 5 MHz operation. The 1.5 MHz comp rate will allow information data rates with useful coding gain up to about 512 Kbps, while the improved 5 MHz device will allow 1.5 Mbps (rates 1/4, 1/2, 3/4) and 3.0 Mbps in code rate 7/8.

1.1.3 Continuous Data Coding Objectives

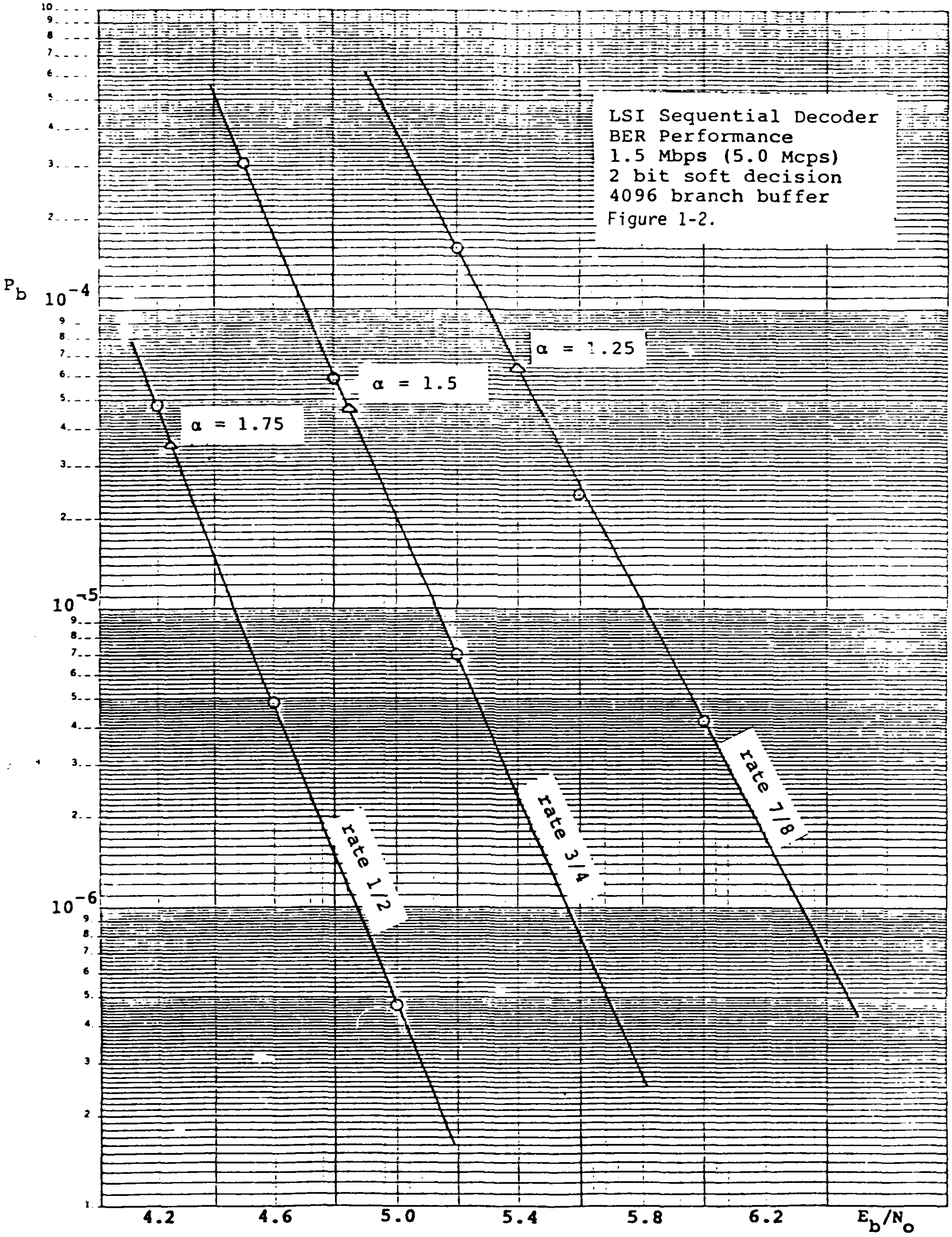
In continuous data systems, BER versus channel error rate is the figure-of-merit relationship for decoders. In this case, BER includes the cumulative effects of buffer failure events and

errors which defeat the code itself. The latter become apparent at code rate 7/8. The expression for output Pr[error] is derived principally from the expression for Pr[Buffer Failure] since this is the dominant effect. Therefore Pr[BF] = k T exp(-a), and output Pr[error] = p L Pr[BF] = p K T exp(-a). In the preceeding, L is average channel transparency window length associated with a buffer failure event and the new constant K = L*k. Since log Pr[error] = log(p) -a log(T) + log(K), this curve approximates a straight line when plotted semi-logarithmically versus E_b/N_o. This is due to the very nearly linear relationship between the Pareto exponent (a) and both log(p) and E_b/N_o for code rates at or above 1/2 and 2-bit quantized AWGN channels. The standard performance plot is log Pr[error] versus E_b/N_o.

Measurements of BER versus Eb/No have been performed on the LS56 model with a computation rate of 1.5 MHz. Figures 1-2, and 1-3 represent LS56 performance against 2-bit soft decision and 1-bit hard decision AWGN noise channels for the 4K-branch buffer size and 300 KBPS data rate. This simulates a 1.5 MBPS data rate with a 5.0 MHz computation clock.

1.1.4 Discrete Block Data Coding Objectives

The relevant statistic in block sequential decoding is the distribution of probability that more than N computation cycles will be required to completely decode a particular size of data block. The interpretation of this statistic allows the system design to provide a known-adequate combination of block-buffering and maximum aggregate data rate for successful decoder operation. Using an exact model, a block length of approximately 1000 bits was used to determine these distributions for the three code rates of 2-bit soft decision mode and the two code rates of hard


LSI Sequential Decoder
BER Performance
1.5 Mbps (5.0 Mcps)
2 bit soft decision
4096 branch buffer
α = 1.25
α = 1.5
α = 1.75
rate 1/2
rate 3/4
rate 7/8
P_b
10^{-4}
10^{-5}
10^{-6}
4.2
4.6
5.0
5.4
5.8
6.2
E_b/N_o
Figure 1-2.

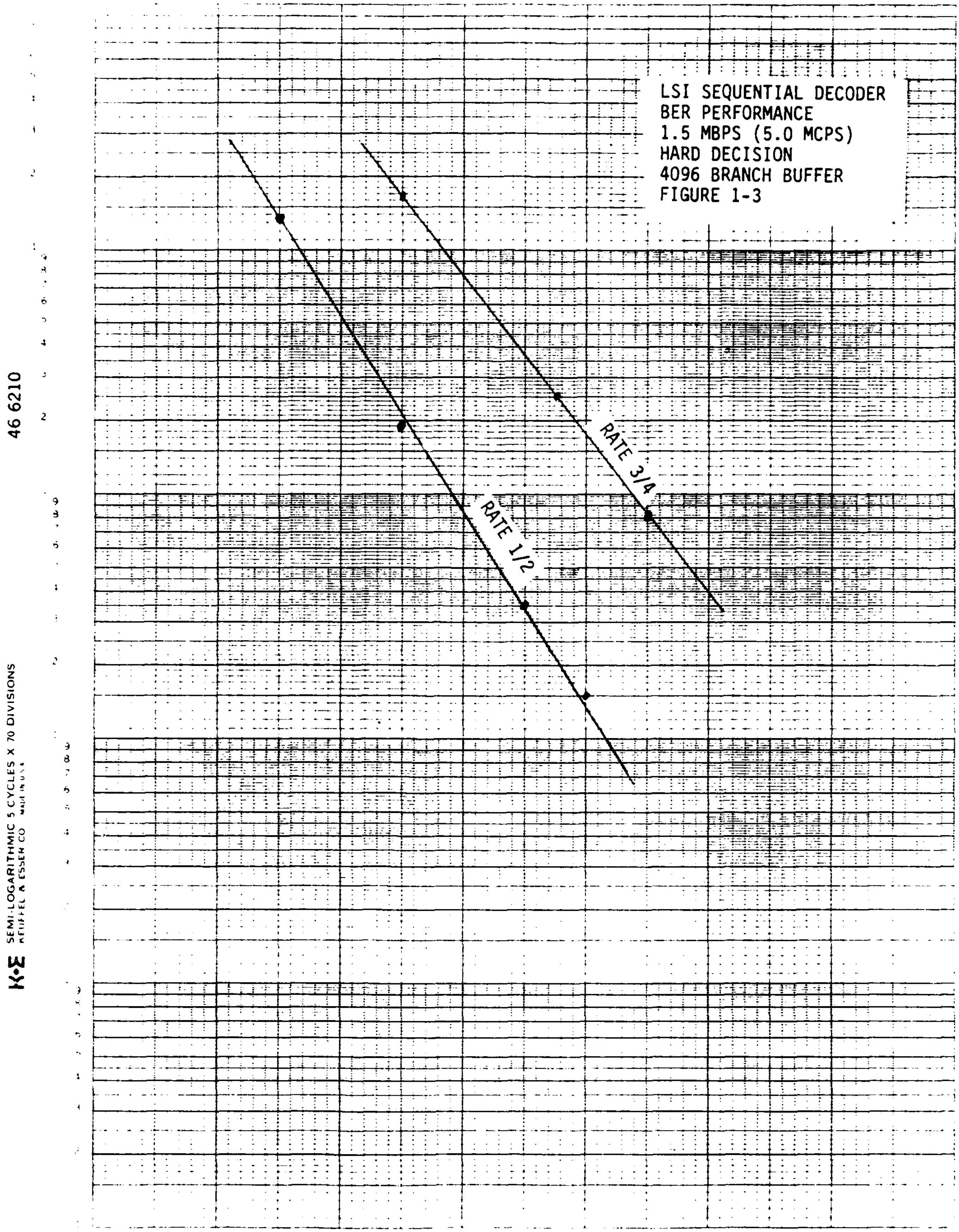

LSI SEQUENTIAL DECODER
BER PERFORMANCE
1.5 MBPS (5.0 MCPS)
HARD DECISION
4096 BRANCH BUFFER
FIGURE 1-3
RATE 3/4
RATE 1/2
46 6210
K•E SEMI-LOGARITHMIC 5 CYCLES X 70 DIVISIONS
KEUFFEL & ESSER CO. MADE IN U.S.A.

Figure 1-4. Rate 1/2 Soft Decision Computation Distributions

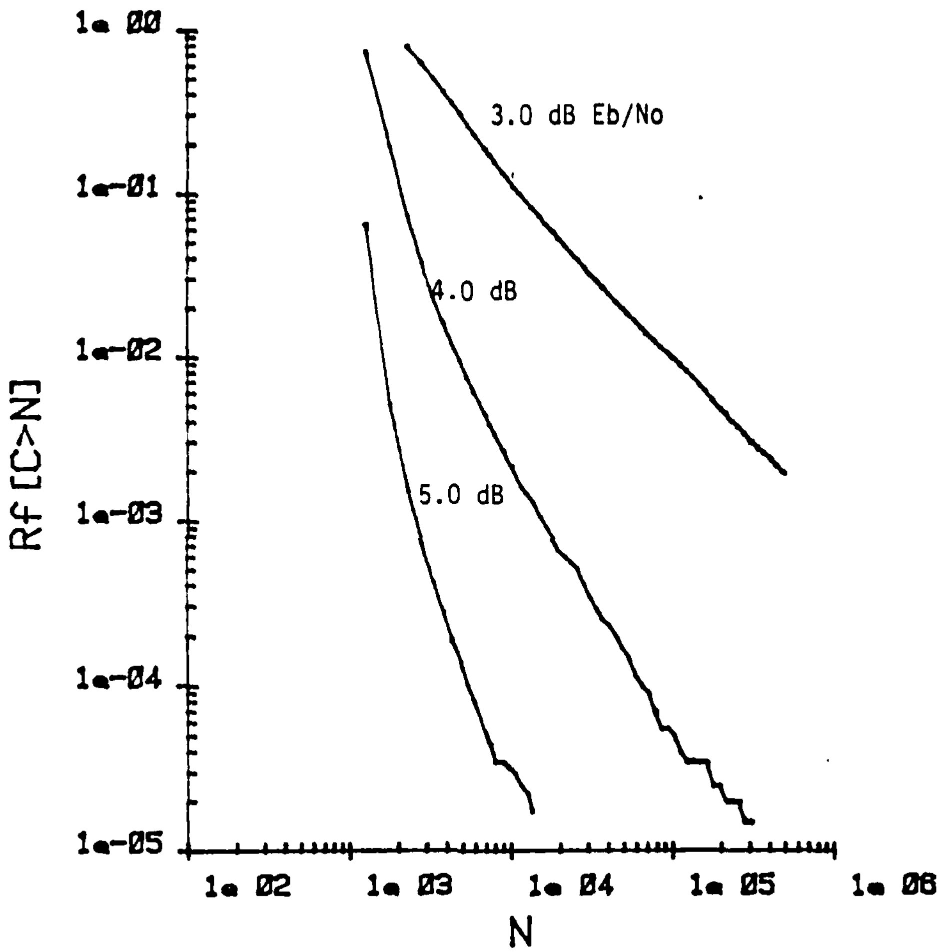


Figure 1-5. Rate 3/4 Soft Decision Computation Distributions

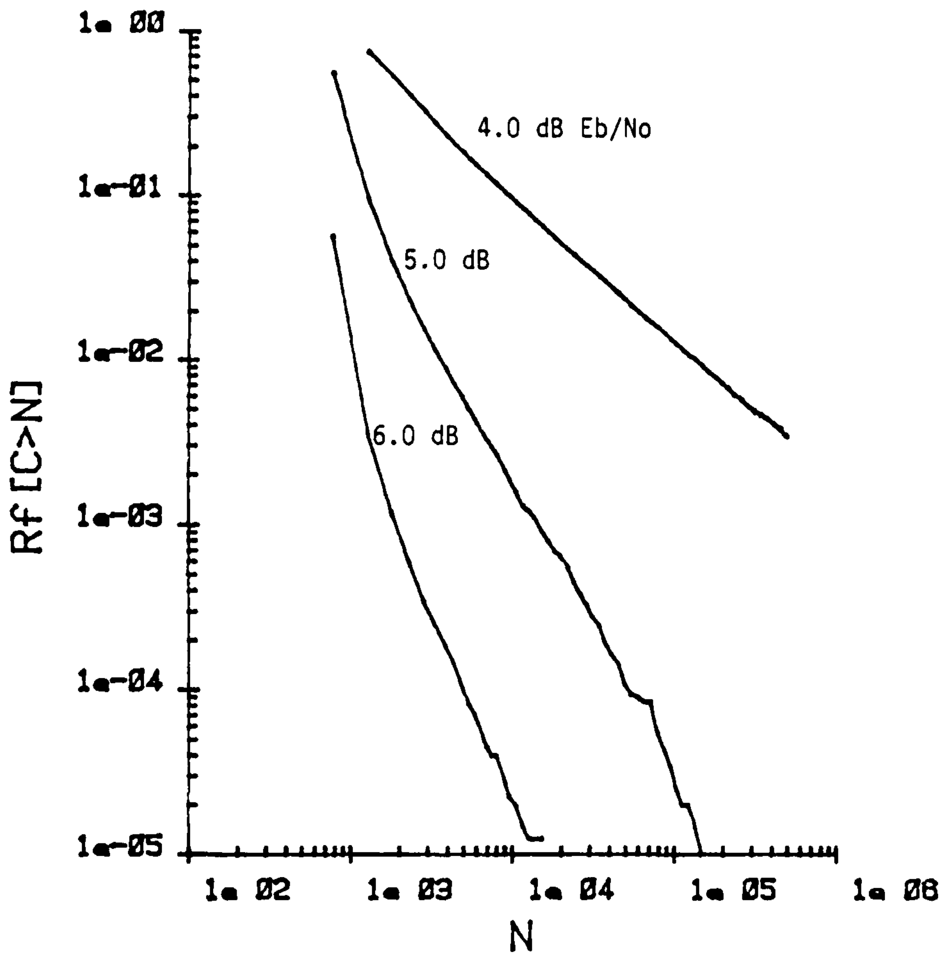
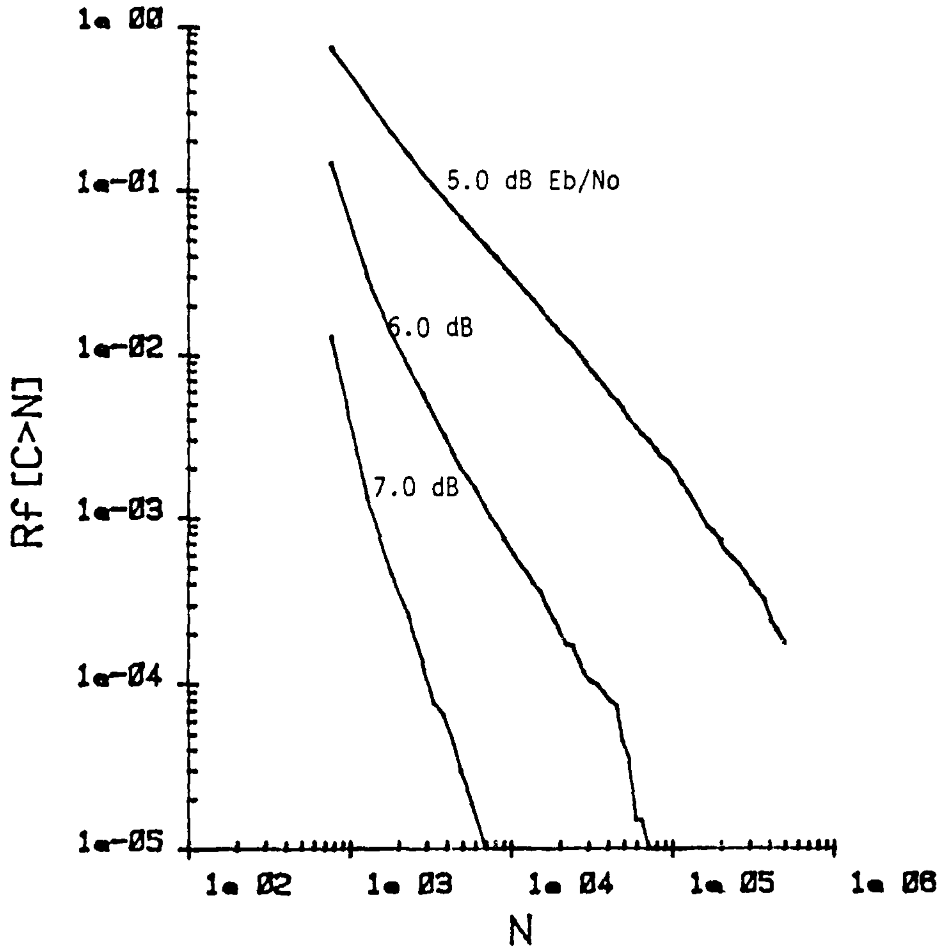


Figure 1-6. Rate 7/8 Soft Decision Computation Distributions



decision. Figures 1-4, 1-5 and 1-6 present soft decision and Figures 1-7 and 1-8 present hard decision. These figures show the probability that a 1024-bit long block will take "N" computation cycles to decode for several E_b/N_o settings. Note that these measures of decoder performance do not depend in any way on the absolute value of computation clock rate.

1.1.5 Contiguous Block Data Coding Objectives

Contiguous blocks are simply blocks in a continuous stream. The discrete block data performance statistic applies in this case with the warning that performance may vary considerably with a mix of data block lengths.

1.2 System Environments

1.2.1 General Coding System Features

The LS56 is designed to provide encoding or decoding functions with minimum additional circuitry. It is possible to use the same LS56 to alternately encode or decode data blocks, but it is not possible to do both operations simultaneously: two LS56 devices are required for full duplex operation in either block or continuous data operation.

The LS56 encoder utilizes a portion of the sequential decoder logic to accomplish convolutional coding. The code characteristics are specified in section 7.1. The LS56 decoder utilizes essentially all of the logic to perform code-compatible decoding.

The complete encoding or decoding system encompasses the

Figure 1-7. Rate 1/2 Hard Decision Computation Distributions

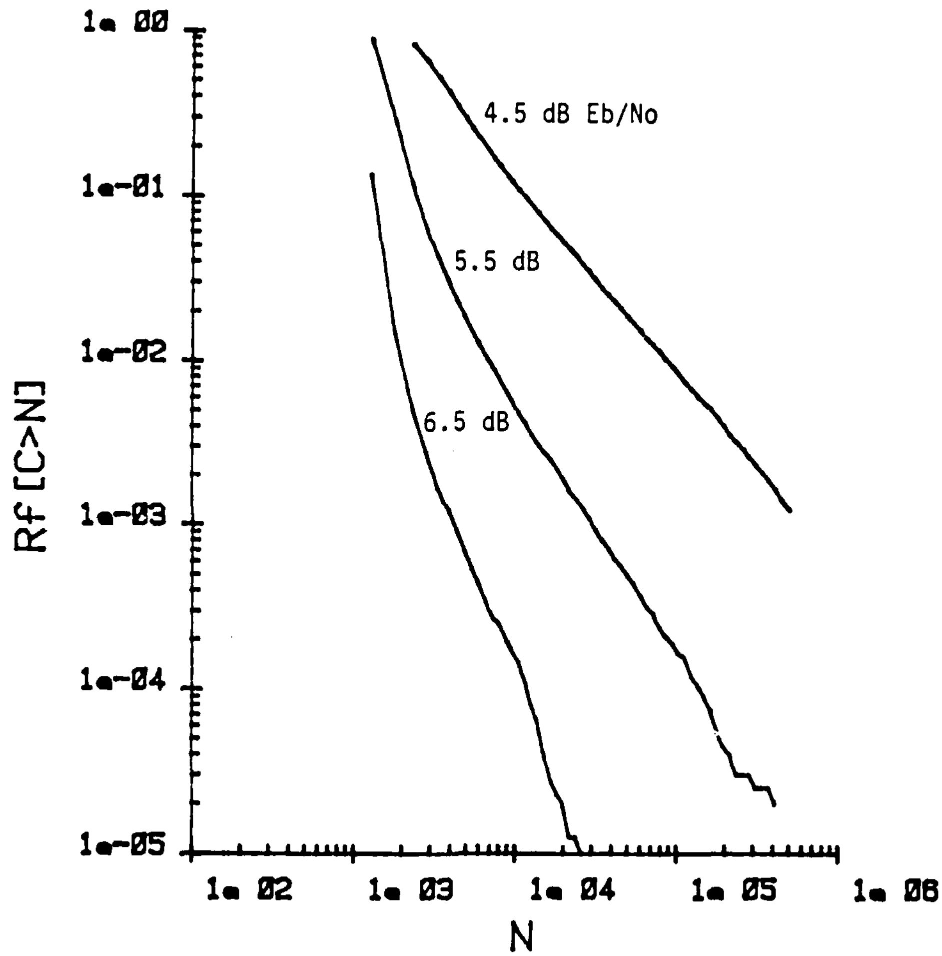
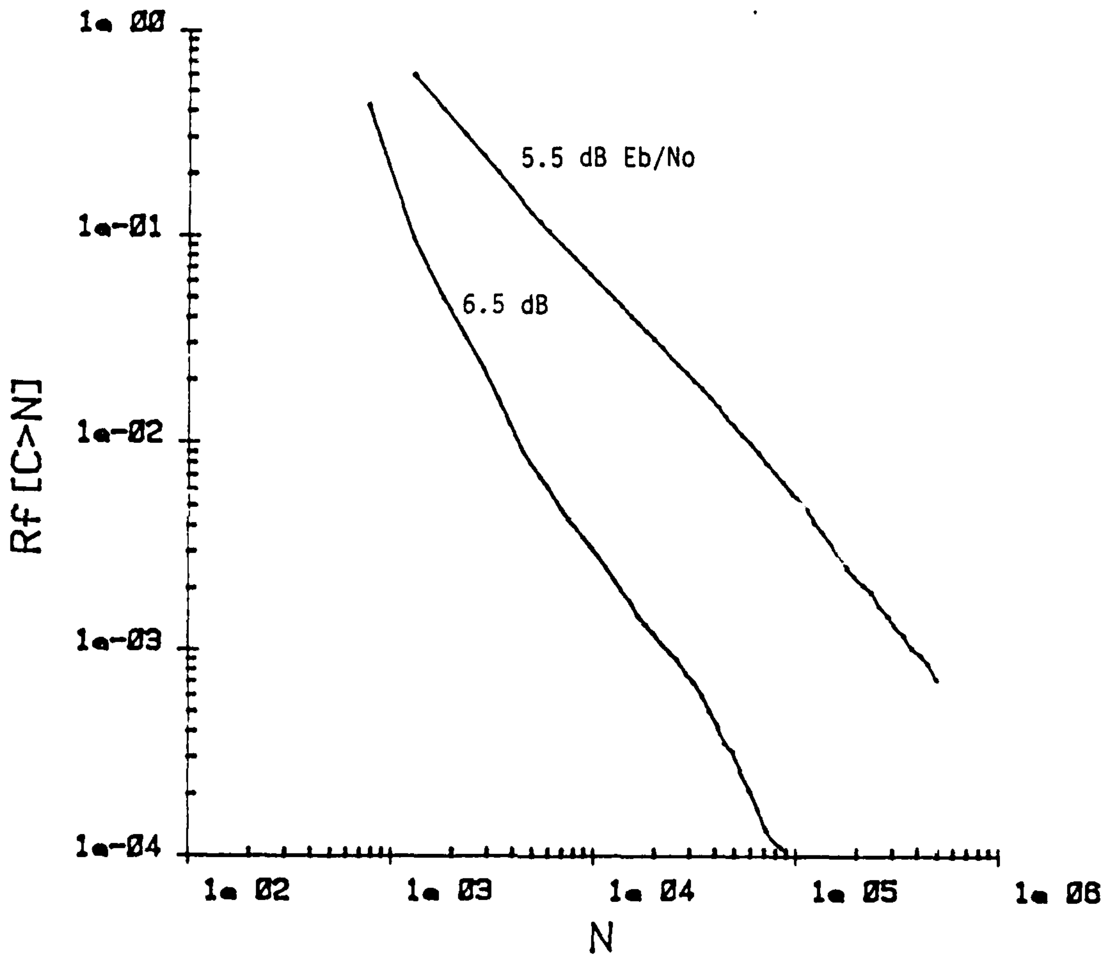


Figure 1-8. Rate 3/4 Hard Decision Computation Distributions



input and output interfaces, clock derivation for normal rate 3/4 or 7/8 coding and RAM memory in the case of decoding. Ancilliary circuits, such as interleaver/deinterleavers or error rate monitors may be optionally included. In order to allow flexibility in application, the LS56 is designed with two primary interface options: block data input/output or continuous data input/output. There are several variations of the continuous data interface.

1.2.2 Continuous Data System Environment

The LS56 may be applied to a range of continuous data systems. These include low data rate, low-cost single channel/carrier links, burst error channels when interleaving is supplied, and military applications. The encoder mode has been designed to operate with a minimum of support hardware (normally only a PLL from which to derive the channel symbol clock). The decoder operation is compatible with either PLL-derived information rate recovery or an internally-derived information rate clock. The decoder can resolve all levels of synchronization associated with BPSK, QPSK or Offset QPSK. The LS56 includes a built-in differential encoder/decoder to resolve the sense of output data in all continuous data modes, regardless of modulation type. The decoder also provides a continuous indication of corrected channel errors to facilitate channel error rate monitoring in an external circuit.

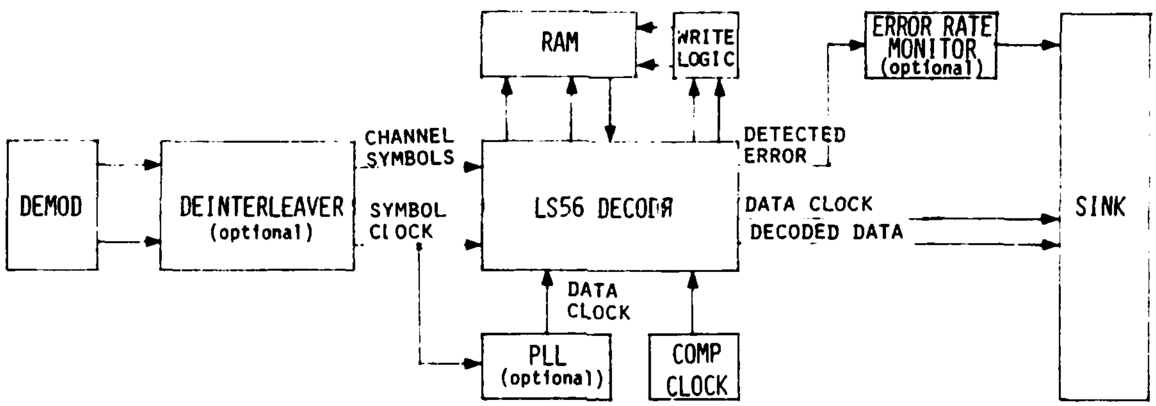
The LS56 latches data in continuous data mode by detecting the falling edge of an input clock which is at the midpoint of the input data transitions. The LS56 outputs continuous data through an internal first-in first-out (FIFO) buffer. This buffer is written via the computation clock and read to the

external interface with either an internally or externally generated clock, depending on the operating mode. The internally generated clock is used in all rate 1/2 decoding, QPSK rate 1/2 encoding, and the option exists in BPSK deocding, rates 3/4 and 7/8. All other modes require an externally generated output clock.clocks for encoding or rate 3/4 and 7/8 decoding. The selection of option is driven by system requirements. In any case, inputs and outputs are always synchronous with the appropriate and externally-accessible bit, branch (2-bit) or symbol clock.

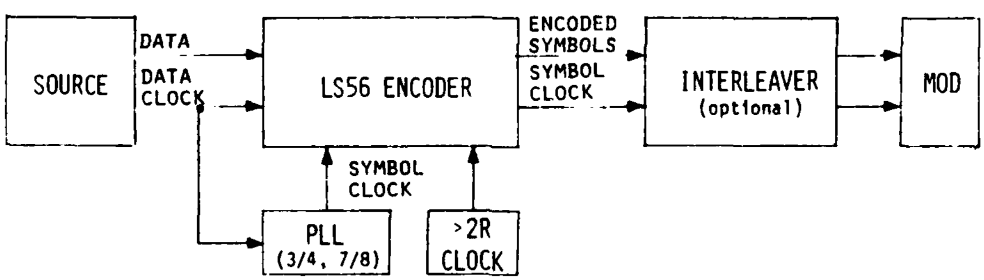
Due to the clock edge detection used in continuous data I/O, there exists an upper input data rate relative to the computation clock rate. There may not exist any data input clock, in encoding or decoding, with rate exceeding 85% of half the computation clock rate. This limitation does not apply to output clock rates in encoding operation.

The block diagrams in Figure 1-9 illustrate the configuration of continuous encoders and decoders built with the LS56. Neccessary to both encoder and decoder systems are an external computation clock. The decoder further requires an external buffer memory and associated Write logic. This external logic may be as few as 3 IC's including RAM for the lower performance applications and about 15 IC's including RAM for higher performance applications. The LS56 has generally been designed to be limited in ultimate performance by the access speed of available RAM memory.

Figure 1-9. General Continuous Data Decoder/Encoder



GENERAL CONTINUOUS DATA DECODER



1.2.3 Discrete Block Data System Environment

The LS56 operates as a peripheral device with memory access in the discrete block case. Typically, direct memory access (DMA) is employed. The decoding system must provide DMA support in addition to the buffer RAM memory always associated with decoding. The encoding system need provide only DMA support. A DMA Interface chip is being developed to augment the LS56 in this mode. The complete DMA-type decoder system also requires a DMA Controller IC matched to the processor/bus type in use. This system is interfaceable to an MC68000 or the 8086-family processors.

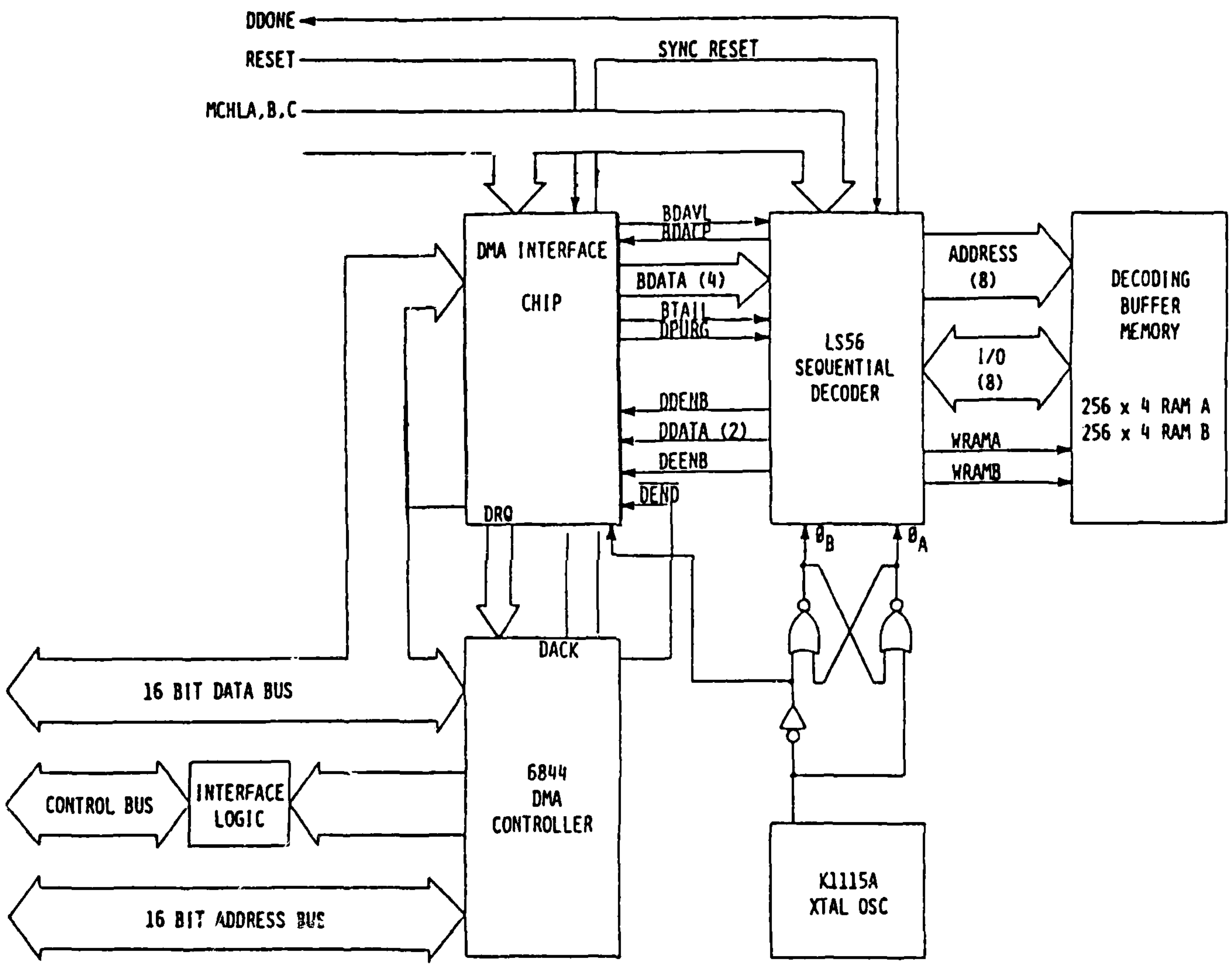
In discrete block mode, the LS56 accepts input data with a pair of handshaking signals, BDAVL/BDACP. The LS56 outputs data using an enable-strobe DDENB with 2 data output lines DDATA,B. All signals in this environment are synchronous with the LS56 computation clock. This description is true of both encoding and decoding.

The diagram in Figure 1-10 illustrates a representative DMA-type decoder system design.

1.2.4 Contiguous Block Data System Environment

The primary differences between discrete block and contiguous block operation are in the manner of buffering pre-decoder blocks while the LS56 performs non-realtime searching within preceding data. In the discrete block case, there is an implicit assumption that newly received blocks occur with a random but significant amount of inter-block spacing which allows relatively slow DMA-type interfaces and reseting of the LS56

Figure 1-10. Packet Radio Sequential Decoder Subsystem



prior to each new block. In contiguous block data, operation is almost equivalent to continuous data decoding, except that the block-tail structure exists between blocks and each consecutive block may use differing code rates. The contiguous block data encoder can be either a DMA-type or, because encoding is less time consumptive, a continuous-input interface scheme.

The contiguous block data interface to the LS56 is identical to the discrete block interface with the additional provision that the mode controls MCHLA,B,C are treated the same as data. Thus, input data to either an LS56 encoder or decoder is tagged with the appropriate code rate in use. Also, the LS56 may operate from block to block without resets, or it may be operated as in the discrete block case for any block at random in a stream of blocks.

1.3 Input/Output Data Organization

1.3.1 Encoding I/O

The LS56 accepts a serial information bit stream for encoding. The distinctions between block and continuous data are two-fold. First, block data encoding requires either a reset, or completion of encoding a previous data block, prior to data inputting. Second, data block inputting must be closed with the appropriate number of tail bits (relative to code rate used) indicated explicitly through the BTAIL input. Continuous data has neither of these conditions imposed.

All encoding data is input to the BDATA terminal. Encoded data output may utilize either DDATA or the pair DDATA,B depending upon modulation and data type. Figure 1-11 illustrates

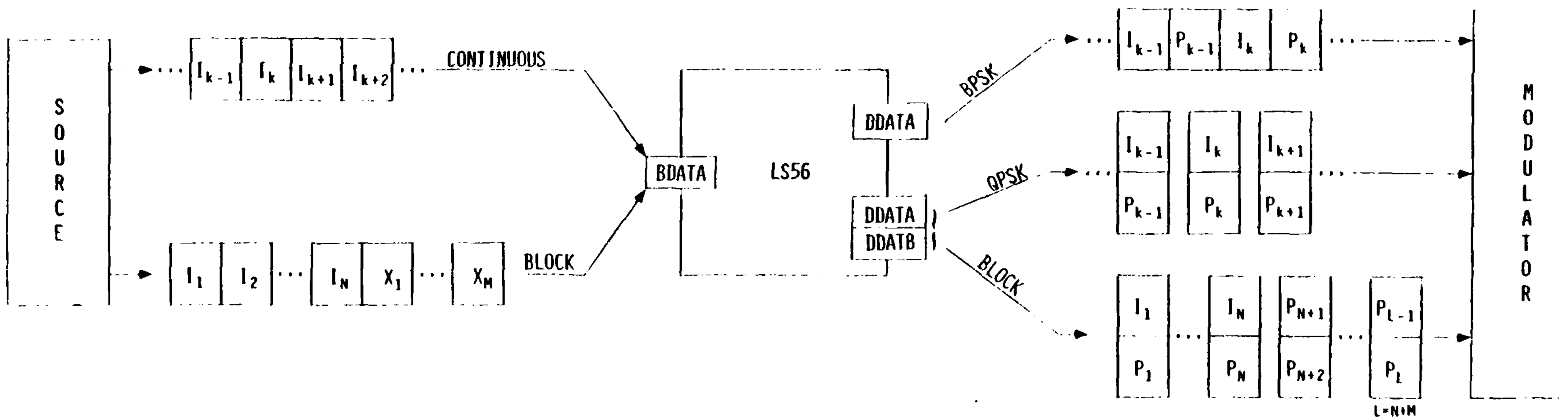


Figure 1-11. LS56 Encoding Data Formats - Code Rate 1/2

I/O progressions for all modulation and data types using code rate 1/2 as an example.

1.3.2 Decoding I/O

The LS56 accepts a variety of input formats depending on modulation and data type. Decoded data output format depends only on data type. The input terminals BDATA-D are divided into two sets for decoding: BDATA,B and BDATC,D. Each set of terminals accepts a single demodulator decision. This decision may be 1-bit or 2-bit quantized (or erasure encoded as shown in section 1.3.11). The output terminals DDATA,B each represent one bit of decoded data (estimates of each original uncoded information). Figure 1-12 illustrates I/O progressions for all modulation and data types using code rate 1/2 as an example.

1.3.3 Encoded Data Organization

Encoded data takes several forms depending on data type and code rate. In block data, the encoder output sequence includes a main body of encoded data pairs and a tail of encoded-parity-only pairs. In continuous data, there is no main body and tail structure.

The fine structure of encoded data depends on code rate only. In rate 1/2, pairs of information and parity bits appear at DDATA,B (except in BPSK encoding where a serial output stream with alternating information and parity appears at DDATA). The higher rate codes are punctured in structure and bear no parity commonality with each other or code rate 1/2. Thus, rate 3/4 encoder output features alternating pairs of information and information,parity bits. Rate 7/8 is similar.

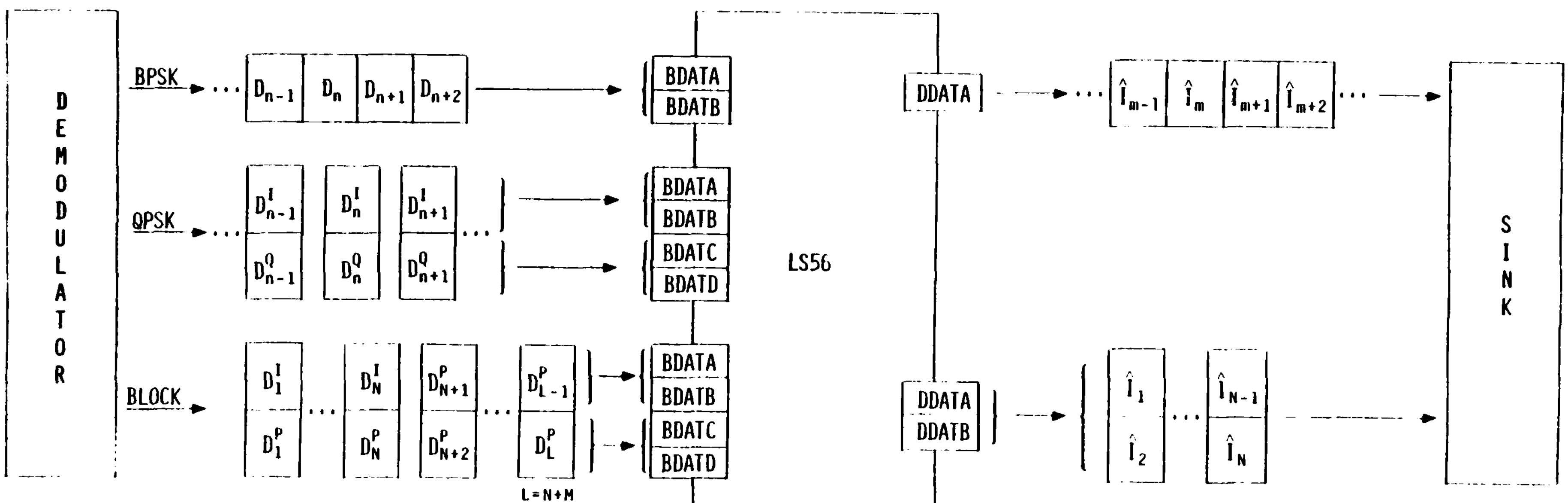


Figure 1-12. LS56 Decoding Data Formats - Code Rate 1/2

The continuous encoded data outputs are shown in Figure 1-13 for QPSK modulation (BPSK encoding is serial output). The block data outputs are shown in Figure 1-14.

Only code rates 1, 1/2, 3/4 and 7/8 may be encoded with the LS56. Rate 1 encoding is for use in contiguous block mode, where all data to be transmitted goes through the encoder system.

1.3.4 Block Encoding Interface Procedure

The LS56 single block encoding procedure may be described in terms of interface transactions. The requisite sequence of interface transactions effectively defines the external interface characteristics at the protocol and timing levels.

The sequence of block encoding interface transactions is flow-diagrammed in Figure 1-15. In this figure, actual LS56 interface signals are capitalized. The encoder mode of operation, defined by MENC and MMODA,B, must be stable prior to LS56 reset (via DRESET) and throughout encoding. Next, assertion and subsequent removal of DRESET may occur for any number of LS56 cycles; complete reset occurs within one cycle. The setting of code rate mode, defined by MCHLA,B,C, may take place prior to or concurrent with the first input transaction of information. Thereafter, the code rate mode is treated identically to information and this implies that code rate mode controls must remain stable throughout encoding. Each input transaction involving BDATA only is a transfer one bit of information data by definition except for block tail encoding which is discussed below.

Encoder information inputs are handshake-driven in the

Figure 1-13. LS56 Encoder Code Rate Sequencing, Continuous Data

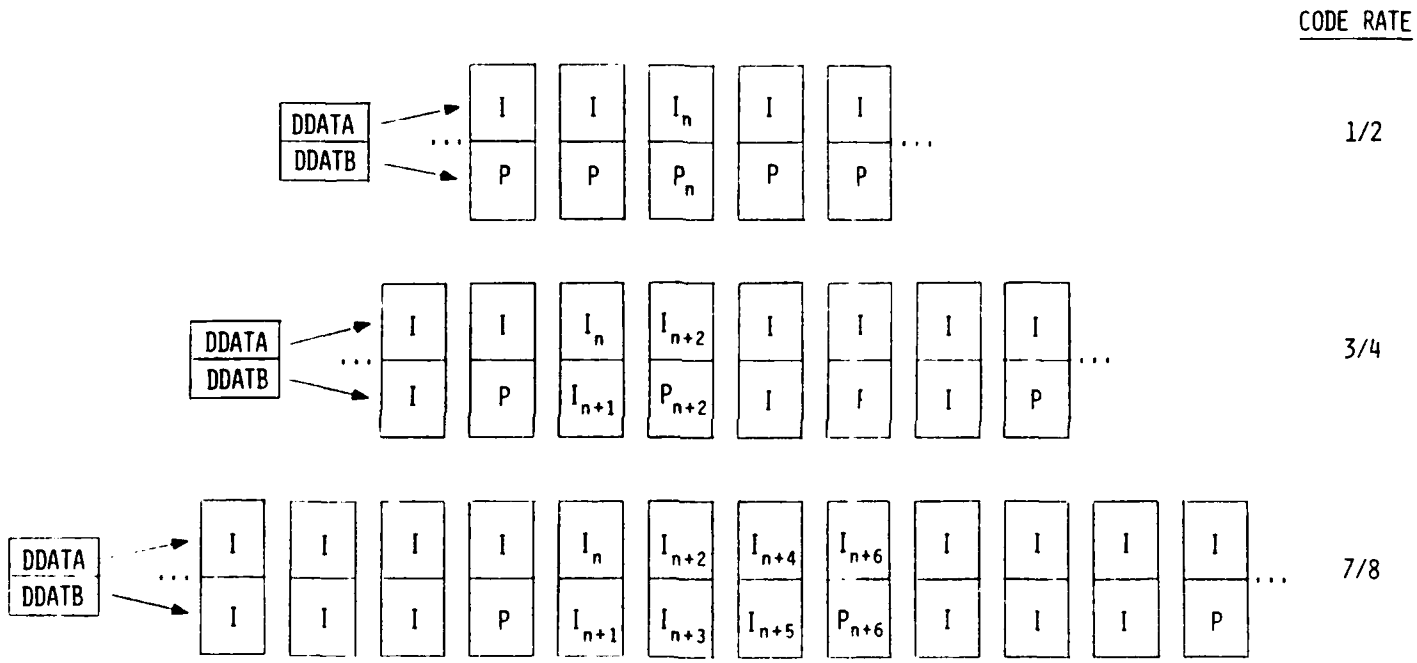


Figure 1-14. LS56 Encoder Code Rate Sequencing, Block Data

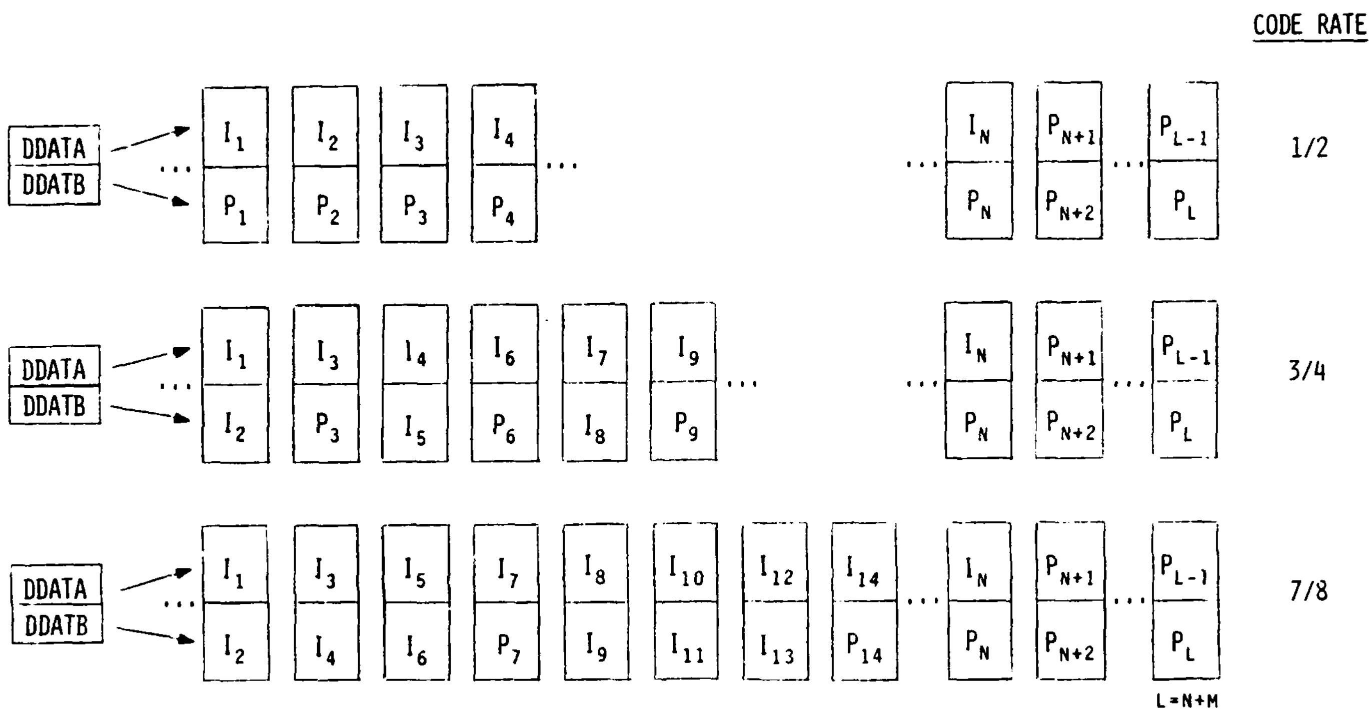
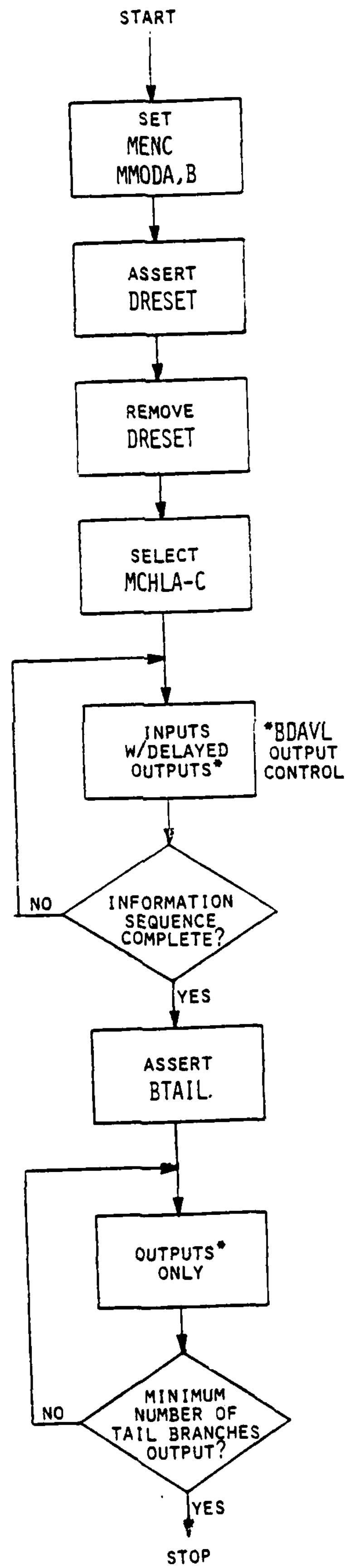


Figure 1-15. Single Block Encoding Flow-Diagram



LS56. This technique allows an external interface of unknown speed and efficiency to be used with minimum control overhead. The handshake consists of the assertion of BDAVL by the external interface indicating an information bit ready for input at BDATA, and a subsequent reply with BDACP by the LS56 indicating acceptance of the information within that cycle. The LS56 will indicate its ability to accept a new information bit by asserting BDACP, regardless of the readiness of the external interface. By definition, input transactions only take place at the handshake: the concurrence of BDAVL and BDACP. The BDACP reply to the assertion of BDAVL can occur in the same cycle.

Encoded data outputs are related to the input transactions and are handshake-driven as well. The output handshake consists of the assertion of BDAVL by the external interface indicating that it is ready for output data at DDATA,B, and a subsequent reply with DDENB by the LS56 indicating that a 2-bit encoded output is available. Again, the DDENB reply may be concurrent with BDAVL. Encoded data outputs at DDATA,B begin to be possible (as allowed by BDAVL) within 3 LS56 clock cycles following the first information bit input transaction. This delay accounts for a short encoded output formatting within the LS56. Outputs do not in general occur periodically except in code rate 1/2 due to fundamental relationships between the number of information bits bits and the total number of encoded output bit-pairs (branches). There will be input transactions not accompanied by output transactions.

The group of input transactions following the setting of MCHLA-C in Figure 1-15 represent the encoding of the block information. This input information sequence is encoded starting

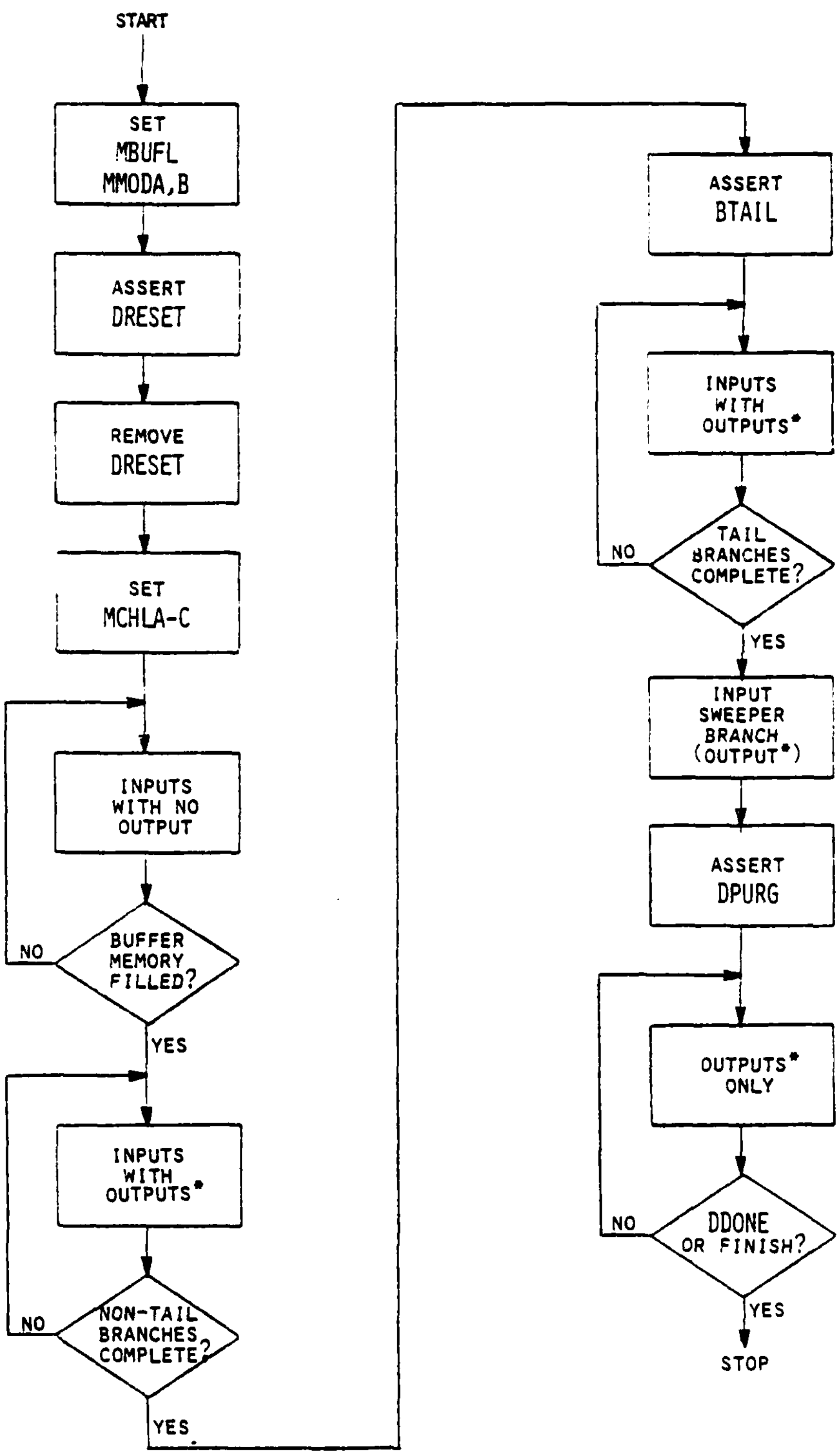
from an all-zeros encoder state (as insured by the DRESET step or properly tailing off a previously encoded contiguous block). When the user information is exhausted by the encoder, the remaining state of the encoder is output as the encoded tail using an all-zeros encoder-input sequence. This final tail sequence is provided internally by the LS56 when the BTAIL input is asserted in encoding. Thus, no actual input transfers are occurring although the BDAVL/BDACP handshaking continues as before. The block tail outputs which exit the LS56 under BDAVL/DDENB handshake control are specially formatted for minimum length (in pairs of bits for each such output). The LS56 will continue to output encoded tail pairs (branches) as long as BTAIL is asserted. Responsibility for determining required tail length per block is assigned to the external equipment. Two tail parity bits are output for every BDAVL/BDACP handshake.

1.3.5 Block Decoding Interface Procedure

The LS56 single block decoding procedure may be described in terms of interface transactions. The requisite sequence of interface transactions effectively defines the external interface characteristics at the protocol and timing levels.

The sequence of block decoding interface transactions is flow-diagrammed in Figure 1-16. In this figure, actual LS56 interface signals are capitalized. The decoder mode of operation, defined by MBUFL and MMODA,B, must be stable prior to LS56 reset (via DRESET) and throughout decoding. Next, assertion and subsequent removal of DRESET may occur for any number of LS56 cycles; complete reset occurs within one cycle. The setting of channel mode, defined by MCHLA,B,C, may take place prior to or concurrent with the first input transaction of branch data.

Figure 1-16. Single Block Decoding Flow-Diagram



*UNDER CONTROL OF BDAVL

Thereafter, the channel mode is treated identically to branch data and this implies that channel mode controls must remain stable throughout decoding. Each input transaction involving BDATA,B,C,D is a transfer of one branch of data by definition except for block tail inputs which are discussed below.

Branch data inputs and outputs are handshake-driven in the LS56. This technique allows an external interface of unknown speed and efficiency to be used with minimum control overhead. The handshake consists of the assertion of BDAVL by the external interface indicating a branch of data is ready for input at BDATA-D, and a subsequent reply with BDACP by the LS56 indicating acceptance of the branch data within that cycle. It is the case that the LS56 will always indicate its ability to accept a new branch of data, by asserting BDACP, regardless of the readiness of the external interface. By definition, input transactions only take place at the handshake: the concurrence of BDAVL and BDACP. In the interest of input speed and efficiency, the BDACP reply to a BDAVL may occur concurrently (within the same cycle) with that BDAVL.

Branch data outputs are related to the input transactions and are handshake-driven as well. The output handshake consists of the assertion of BDAVL by the external interface indicating that it is ready for output data at DDATA,B, and a subsequent reply with DDENB by the LS56 indicating that a 2-bit output is available. Again, the DDENB reply may be concurrent with BDAVL in the interest of maximizing output rate. Decoded data outputs at DDATA,B begin to occur when the LS56 has either decoded one complete buffer memory length of branch data or decoded an entire block shorter than the buffer memory in length. In the latter

case, output arises from the DPURG protocol discussed below. In the former case, outputs occur regularly with the occurance of inputs, although irregularly in absolute time due to the presence of decoder searches in between acceptances of new data and, hence, opportunities for outputs. Outputs do not in general occur periodically except in code rate 1/2 due to fundamental relationships between the number of input branches and the total number of encoded information bits. There will, in general, be input transactions not accompanied by output transactions.

The first group of input transactions following the setting of MCHLA-C in Figure 1-16 begin to fill the buffer memory as these branches are tentatively decoded; sequential decoding will characteristically amend some of these decoding decisions as new channel data inputs arrive. After a decoding decision is one buffer memory length "old", this decision is irrevocably fixed and may therefore be output. The earliest inputs are not, therefore, accompanied by outputs until the buffer memory is full.

The second group of input transactions are accompanied by outputs as the "old" decisions are removed from buffer memory so that the new channel branch data may be written over the old data. This phase of input/output (I/O) continues until all branch data associated with useful encoded information has been input. Following this data are branches associated with known encoder data required to sequentially transfer the final information-driven state of the encoder across the channel; these last data branches are called "tail" branches.

As the final group of input transactions occur, the BTAIL

input variable must be asserted concurrent with BDATA-D. This phase of I/O causes a special input construct, internal to the LS56, to exist in which each input tail branch actually conveys the equivalent of 2 ordinary branches to the decoder. Although no special external interface considerations accrue from this, it is the case that decoder response to BDAVL via BDACP will occur twice as slowly, on the average, as in the non-tail portion of the block. Fortunately, the tail portion of block is very short in branch-count compared to the useful non-tail portion. Block tails are, in any event, essential to consistent decoding performance across the entire block.

The LS56 design creates the requirement that, following the input of all received tail branches, a "sweeper" branch be input to the decoder. This "sweeper" branch is a construct of the external interface and is not passed through the channel. The sweeper branch consists of ZERO sign data and ONE magnitude/erasure data, which when input to the LS56 guarantees a "forward" decoding "move". Such behavior is necessary to insure that any decoding search behavior arising from the final tail branch is completed prior to entrance to the DPURG operation or the start of the next contiguous block. In contiguous block mode, the first branch of the new block can be input after the "sweeper" tail input.

The DPURG operation consists of undefined input transactions and associated, useful output transactions. The DPURG variable is asserted by the external interface following acceptance of the sweeper branch; then, enabled as always by BDAVL, the final decoded output data residing in the buffer memory will be output. The buffer memory is, in essence,

"purged" of its content of valid decisions, for which there are no further useful inputs. The DPURG operation disallows decoding searches and facilitates the most rapid output of the end of block as allowed by the external interface. For blocks possessing length less than the buffer memory length, all outputs will occur as a result of the DPURG operation.

At the conclusion of a buffer memory purge, the LS56 will signal the external interface with the DDONE status output. This status signal may be used as a consistency check by the interface or as an actual control over outputting.

1.3.6 Block Data Format Restrictions

There are two restrictions in block data required by the LS56 decoder (and hence, encoder). First, there must be an integral number of branch-groups in any encoded main body. A branch-group is a cycle of encoded data with it's associated parity bit (see Figure 1-14). A branch-group is one pair in length for rate 1/2, two pairs for 3/4 and four pairs for 7/8. Thus, rate 1/2 blocks may contain any number of main body pairs; rate 3/4, a multiple of two pairs and rate 7/8, a multiple of 4 pairs.

Second, there exists a minimum length of the block data tail acceptable in decoding. This length is 18 pairs of encoded parity bits in rate 1/2; 11 pairs in rate 3/4, and 7 pairs in rate 7/8. This minimum tail length includes one pair of additional ZERO,ZERO tail data at the end of the block in order to insure that the decoder will advance properly over a contiguous block boundry or into the purge operation. A block data tail may always be extended with ZERO,ZERO pairs to allow

filling of a given block length.

1.3.7 Block Data Tail Inputs

In encoding, a special provision has been made to compress the number of inputs required to acheive a block tail. When BTAIL is asserted, a complete branch-group of ZERO's is processed in the LS56 encoder resulting in a single parity bit ready for output every other processing step (since outputs are in pairs). This feature allows the formation of block tails in 36, 22 and 14 input cycles for rate 1/2, 3/4 and 7/8 respectively.

In decoding, the assertion of BTAIL concurrent with BDATA-D allows direct decoding of pairs of parity bits as generated in the encoder; no reformatting is required.

1.3.8 Block Data Tail Output

As indicated in Figure 1-12, decoded block data output from the LS56 includes only the main information body of the block. Decoded tail ZERO's are not output.

1.3.9 Continuous Encoding Interface Proceedure

The encoding interface consists of a serial data input at BDATA with an associated data rate clock input at BDAVL. Due to a logic error in the LS56, the data input to BDATA <u>must</u> be latched externally to the LS56 with a flip-flop clocked in the encoder computation clock. Additionally, the data <u>must</u> change on falling edges of the data rate clock input at BDAVL. This protocol is followed in BPSK or QPSK encoding. The input data is latched on the falling edge of the BDAVL clock and input to the encoder. The encoded data is output at DDATA,B, synchronous with

the encoded data baud clock, output at DDENB/. Data is clocked on the falling edge of the output clock and should be captured on the rising edge of the output clock by the external interface. The encoder baud clock is internally generated in QPSK rate 1/2, and must be externally generated for all other encoding modes.

The BPSK mode output is a serial interface, data output occuring at DDATA. The encoder output clock, input at DECLK, must be phase-locked to the input data clock. The ratio of output clock to input clock must be 2, 4/3, and 8/7 for rates 1/2, 3/4, and 7/8 respectively.

The QPSK mode output is a two-bit parrallel interface at DDATA,B. The encoder output clock will be different, depending on the code rate. Rate 1/2 encoded output is accomplished with the input data clock, output at DDENB. The rate 3/4 and 7/8 output is clocked with an externally generated clock, phase-locked to the input clock. The ratio of output clock to input clock must be 2/3 and 4/7 for rate 3/4 and 7/8 respectively.

1.3.10 Continuous Mode Decoding Interface Proceedure

The continuous mode decoding interface is varies, depending on the decoder mode, BPSK, QPSK, and OQPSK. All the interfaces input data, synchronous with a baud clock at the BDATA-D inputs, and output data, synchronous with the data rate clock, serially at the DDATA output. The baud clock is input at BDAVL and data is latched internally on the falling edge of the baud clock. The data rate clock is phase-locked to the baud clock and is input at DECLK when necessary. The output data is clocked on the falling edge of the data clock.

In BPSK mode, the data is input at BDATA,B. The data is clocked into the decoder on the falling edge of the BDAVL clock. The data output can be accomplished in one of three ways, depending on code rate and the status of the input variable, DPUNC. First, in rate 1/2, the data output is accomplished with an internally generated baud clock divided by two. This clock is output at DDENB/ for use by the external interface. Second, the output may be clocked with an externally generated data rate clock, phase-locked to the baud clock for rate 3/4 and 7/8. This clock is input at DECLK and the output clock to input clock ratios must be 3/4 and 7/8 for rate 3/4 and 7/8 respectively. Third, in rate 3/4 and 7/8, with DPUNC set to 1 the output is clocked with a punctured version of the baud clock, eliminating the need for an external phase-locked loop. This punctured clock is generated by removing one rising and falling edge every four or eight inputs in rate 3/4 and 7/8. This clock does not have a 50% duty cycle and is output at the DDENB output.

In QPSK mode, the data is input at BDATA-D, with the I channel decision intput at BDATA,B, and the Q channel decision input at BDATC,D. The QPSK baud rate clock is input at BDAVL and data is latched internally on the falling edge. The output is clocked in one of two ways. In rate 1/2, the input baud clock is the same rate as the output data clock so the output data is clocked with a buffered version of the input clock. In rates 3/4 and 7/8, the output clock must be generated externally and be phase-locked to the input baud clock. The ratio of output clock to input clock must be 3/2 and 7/4 for rate 3/4 and 7/8 respectively. In both cases, the output clock is output at DDENB and the data is output at DDATA.

In OQPSK mode, the decoder has the capability of pairing one of two I channel inputs with a Q channel input. It is adviseable to externally latch the input data using the rising edge of the Q channel data clock, assuming the Q channel data changes on the rising edge of this clock. This latched data is then input to the BDATA-D inputs of the decoder and the Q channel data clock is input to BDAVL. The data output is accomplished in the same way as in QPSK mode.

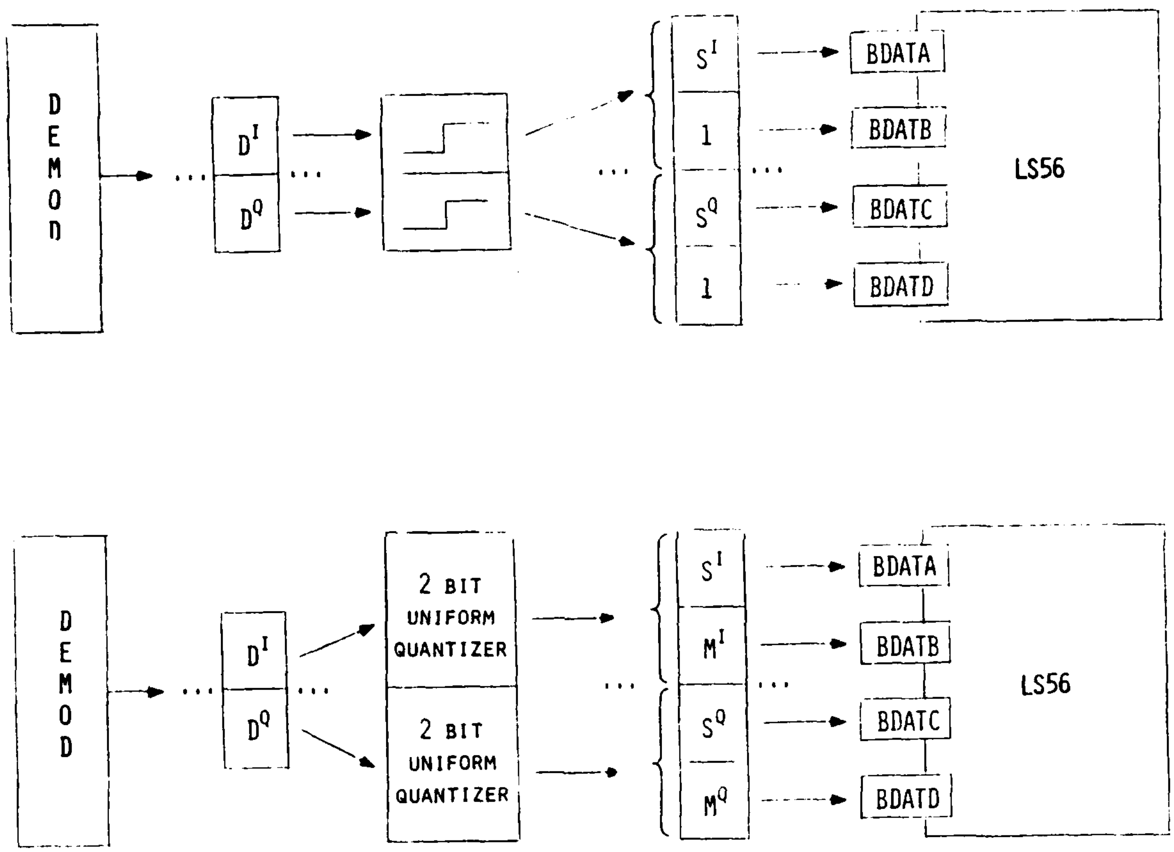
1.3.11 Decoder Interpretation of Channel

The LS56 decodes against a number of channel models. Principal among these are the 1-bit Binary Symmetric Channel BSC, the 2-bit AWGN channel and the 1-bit Binary Symmetric Erasure channel (BSEC). Each of these channels depends upon memoryless noise properties from each channel symbol to the next.

The LS56 interpretation of BDATA-D for the AWGN channels is specified in Figure 1-17. In this figure, "S" is a demodulator sign bit, "M" is a demodulator magnitude bit and "1" is a logic ONE. Generally, the 1-bit BSC channel is similar to the 2-bit AWGN channel except that a ONE must appear in place of estimate magnitude bits. The two cases in Figure 1-18 are shown for QPSK (or Offset QPSK); in BPSK, only the top ("I") data leading to BDATA,B is utilized.

The inclusion of BSEC interpretation of channel allows the LS56 to handle duplicates of a single original block of transmitted data with a decoding advantage. Using this feature, it is possible to construct a rate 1/4 code, otherwise impossible. The rate 1/4 code is actually a semi-repetition code (rate 1/2 repeated once). The properties of the resulting code

Figure 1-17. Input Data Format



approximate a true rate 1/4 code sufficiently to justify the label. The BSEC interpretation is also possible with rate 3/4, but not rate 7/8.

The generation of the pseudo-rate 1/4 code via BSEC is shown in Figure 1-18. Here, a demodulator operates as a 1-bit BSC device with memory of decision history. A channel symbol pair received at time T_1 is retained and compared with a subsequent reception at time T_2. If the symbols match, a logic ONE is sent to the appropriate input, BDATB or D. If not, a logic ZERO is sent and an erasure is thereby declared. The LS56 may operate against this model for virtually any type of modulation or data provided synchronized memory is available.

1.4 Power Supply Requirements

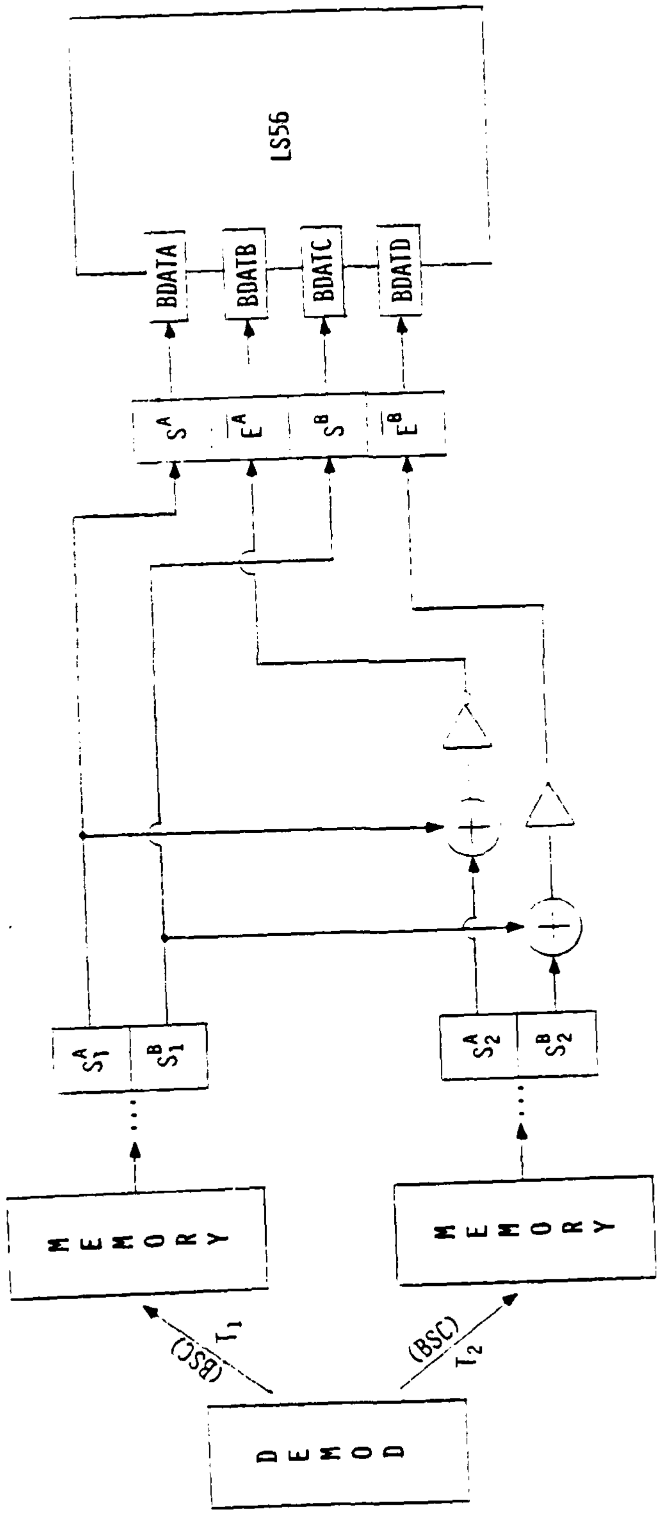
1.4.1 Main Chip Supply

The LS56 is designed to operate with a single +5VDC ±5% power supply. Maximum current required is 250mA over the entire operating range. No substrate bias voltages or special clock voltages are required for normal operation (although special clock voltages will be allowed for operation beyond the specified maximum computation clock frequency of 1.5 MHz).

1.4.2 Computation Clock Voltages

The LS56 is designed to normally operate with a computation clock which provides a maximum low level of 0.2V and a minimum high level of 4.8V. It shall be possible by design to operate at computation clock rates above 1.5 MHz through the use of a high-voltage clock level, nominally expected to be 7.0V; in this case,

Figure 1-18. D[illegible]C, Pseudo-Rate 1/4 Input Data Format



the maximum low level shall remain unchanged.

1.5 *Physical Requirements*

1.5.1 Die Size

The LS56 die has a total area (pads inclusive) of 68,244 sq. mil. This area is defined by die dimensions of 282 mil X242 mil. The approximate active device count of 12,000.

1.5.2 Packaging

The LS56 die shall be packaged one die to a carrier using, at least, a 68-pin ceramic leadless chip carrier with .050" lead spacing. The leadless chip carrier is intended to be socketed in any application. The LS56 may later be packaged in a pinned square package with the same PC board footprint as the leadless chip carrier socket. This specification does not limit application of other carriers as they may become desirable, but rather limits the minimum compatibility of the LS56 die to existing carriers.

1.6 Semiconductor Technology

1.6.1 Device/Material Technology

The LS56 shall be designed for fabrication with n-channel MOS LSI devices on a single substrate. These devices shall utilize silicon gate design and allow for the existence of up to three coincident layers of interconnection as required.

1.6.2 Device Scaling, Feature Size and Topological Rules

The LS56 shall be designed for fabrication using a minimum feature size of 5 microns, as is consistent with manufacturer's definition of topological rules and fabrication technology. The LS56 shall be designed using topological rules which specifically allow for future device size down-scaling to attain higher speed performance.

1.6.3 Typical Device Performance by Design

The LS56 shall be designed consistently with the assumption that highest-performance 2-input, 2-output-load NOR gates with minimal output line capacitance consistent with practical application in the circuit shall typically acheive propagation delays of 5 ns when process parameter variation is within 1 sigma deviation of ideal parameters.

2. INTERFACE SIGNALS

2.1 Operation Control Signals

2.1.1 Clocks

The LS56 shall receive 2 buffered, clock phases of nominal symmetry which possess transitional pairwise overlap. The logical HIGH voltage level of each phase shall be 4.8VDC minimum. The maximum logical LOW voltage level of each phase shall be 0.4V. Each clock phase transition shall possess a 90% V(HIGH) to 10% V(HIGH) (and vice versa) maximum transition time of 30ns. The maximun allowable clock overshoot is 0.3VDC above the high clock level. The maximum allowable undershoot is -0.3VDC relative to the power supply ground. It is advisable to use a specialized clock driver capable of meeting these specifications.

These clocks, CKPHA and CKPHB, shall be of 50 ±10% duty cycle and shall possess pairwise transitional crossover in the voltage range of 0.4v to 1.0VDC. In higher performance applications, the crossover of the clock should be as close to the low level of the clock itself. The acceptable minimum timing features of CKPHA and CKPHB are shown in Figure 2-1.

2.1.2 Reset

The LS56 shall receive a buffered, positive-sense reset input signal synchronous with CKPHA. The maximum required duration of DRESET (at logical HIGH) is one CKPHA cycle in block operation and one output clock cycle in continuous operation. The timing for DRESET to reset all CKPHA-clocked flip-flops is shown in Figure 2-2. The DRESET input meets type A standards

Figure 2-1. Computation Clock Timing

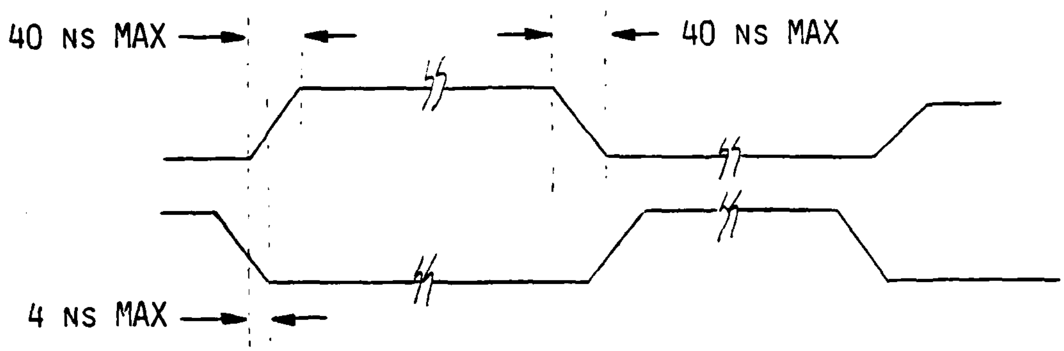
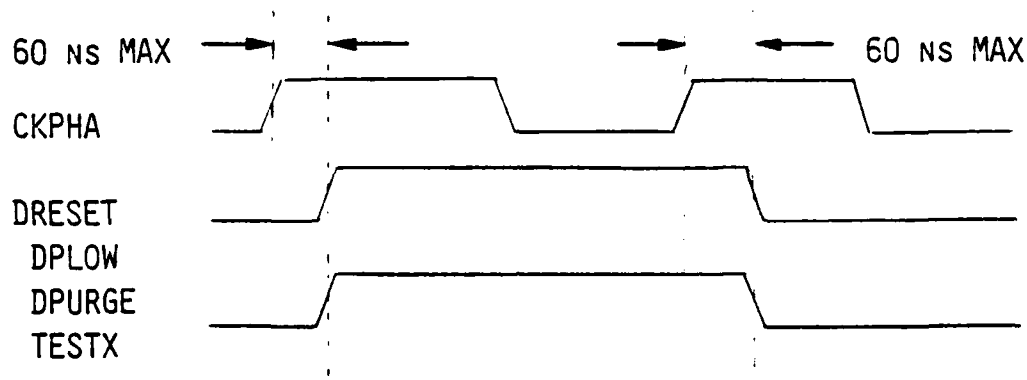


Figure 2-2. RESET, DPLOW, DPURGE, and TESTX Timing



(see section 6.1).

The function of DRESET is to force the LS56 to a completely-known state for the purpose of a) starting the decoding or encoding of a block, b) aborting a block under decode, or c) starting a repeatable series of computations for the purpose of factory testing or self test in any mode. A reset of CKPH-clocked flip-flops may be accomplished in one computation cycle. However, not all internal CKPH-clocked flipflops possess clear inputs reacting to DRESET. The continuous mode output counters are clocked on the output clock and are reset on the rising edge of that clock when reset is active.

The reset places all syndrome registers, path metric counters, and the buffer addressing counters into the zero state. The reset places the branch timing, arithmetic registers, and synchronization logic subfunctions into the correct unique states for beginning block decoding with the next input branch data transfer. The branch data and correction pipelines along with the continuous mode input registers are not reset, as their states may be determined by a sequence of inputs. The Continuous mode output counters are reset, to a unique state for testing of the output buffer.

2.1.3 Decoder Plow-Forward (DPLOW)

In block mode, the LS56 will receive a buffered, positive-sense plow-forward control signal DPLOW synchronous with CKPHA. The DPLOW control input meets Type A standards (section 6.1). The timing of DPLOW is given in Figure 2-2.

DPLOW provides the decoder system with a search control.

The effect of the DPLOW control input, while in normal decoding, is to force the decoder to move forward without searching or correcting data. The DPLOW control may be used in normal decoding in block or continuous mode to cause the decoder to move forward without making corrections. In block mode rate 1 decoding, DPLOW must be asserted to force the decoder forward.

The DPLOW input has added control while the LS56 is in TESTX mode (see section 2.7.1). DPLOW used in conjunction with DPURG can force the decoder to move forward, backward, and sideways depending on the state of DPLOW and DPURG. See table 2-0 for the test mode control assignment.

2.1.4 Decoder Buffer-Purge (DPURG)

In block mode, the LS56 will receive a buffered, positive-sense decoding-buffer purge control signal DPURG synchronous with CKPHA. The DPURG control input signal meets Type A standards (section 6.1). In continuous modes, the DPURG control input is not used by the LS56. Timing for DPURG is shown in Figure 2-2.

The function of DPURG is the initiation of a complete purge (read-out) of the decoding buffer memory for the purpose of completing the outputting of a decoded block when no new input branch data (i.e., the next block) is available to automatically "push" the remainder of the fully-decoded block out. In operation, DPURG is asserted (to logical HIGH) by the external input/output interface after the final Tail branch pair input has been accepted by the LS56: (this final branch of Tail data is actually one complete branch pair over and above the actual Tail data transmitted through the channel.) Once asserted, DPURG may assume any logic state until the buffer purge is complete as

Table 2-1. DPLOW/DPURG Test Mode State Table

<table>
<tr><th>MCHLC,B,A</th><th>CODE RATE</th><th>CHANNEL TYPE</th><th>APPLICABILITY</th></tr>
<tr><td>000</td><td>1</td><td>all</td><td rowspan="4">In Decode Mode:
With All
Combinations
Of Controls
MMODA
MMODB
MBUFL</td></tr>
<tr><td>001</td><td>1/2</td><td>BSC</td></tr>
<tr><td>010</td><td>1/2</td><td>BSEC</td></tr>
<tr><td>011</td><td>1/2</td><td>2 Bit SOFT</td></tr>
<tr><td>100</td><td>3/4</td><td>2 Bit SOFT</td><td rowspan="4">In Encode Mode:
With All
Combinations
Of Controls
MMODA
MMODB</td></tr>
<tr><td>101</td><td>3/4</td><td>BSC</td></tr>
<tr><td>110</td><td>3/4</td><td>BSEC</td></tr>
<tr><td>111</td><td>7/8</td><td>2 Bit SOFT</td></tr>
</table>

signalled by the appearance of the decoder status signal DDONE (see section 2.5.3). Note: An automatic reset of the LS56 is performed internally at the conclusion of all buffer-purges allowing the immediate input of the next data block following DDONE.

DPURG has added control over the LS56 in TESTX mode(see section 2.7.1). Control over the DPLOW and DPURG inputs can force the decoder to move forward, backward, and sideways. See table 2-0 for the state table of DPLOW and DPURG in test mode.

2.2 Mode Control Input Signals

2.2.1 Mode Encode/Decode (MENC)

The LS56 shall receive a buffered, positive-sense control input signal MENC which defines chip mode as either Encode (logical HIGH) or Decode (logical LOW). The MENC control input meets Type A standards (section 6.1). MENC is nominally a static input to the LS56.

The function of MENC is to select between the input/output timing and syndrome decoder operation normally associated with sequential decoding, or to select a different input/output timing and move-forward-only syndrome decoder operation associated with encoding. This dual set of modes allows the LS56 to efficiently replace custom encoder hardware or firmware.

2.2.2 Mode Channel A, B, C (MCHLA,B,C)

The LS56 shall receive 3 buffered, positive-sense input control signals which collectively define code rate and channel type. These three inputs, MCHLA,B,C, shall be continuously present in continuous modes, while they may be synchronously (in CKPHA) valid in block mode operation. The MCHLA,B,C signals meet Type A standards (section 6.1).

The function of MCHLA,B,C, is to provide the basic definition of encoding or decoding mode. Information about code rate (for encoding and decoding) and channel type (for decoding only) is defined according to the matrix of Table 2-2. The codes utilized for each code rate are presented in section 9.1. The channel types acceptable for decoding are illustrated in Table 2-3.

In block mode, the MCHLA-C control group may be set synchronously with CKPHA to allow dynamic code rate switching in "contiguous-block" applications such as packet modems. The MCHLA-C group must be valid concurrent with the first branch of data from the new block and remain stable throughout the decoding process for that block. The LS56 uses the MCHLA-C group to determine the initial timing state of the decoding process for processing the first received branch of channel data in the next input transaction. Dynamic timing of MCHLA-C is identical to that for input branch data (BDATA-D) in block mode. See Figure 2-3 for MCHL A-C timing..

MMODB, A	MODULATION OR FORMAT	SYNC AMBIGUITY	MBUFL	BUFFER RAM ADDRESS SIZE
10	BPSK	2 States (ab) (bc)	0	1024
01	QPSK	2 States (ab) ($\bar{b}$a)		
00	Single Clock OQPSK	2 States (a'b) ($\bar{b}$a)	1	4096
11	BLOCK	None	0	128
			1	256

Table 2.2. LS56 Modulation/Format Mode Controls

2.2.3 Mode Modulation/Format A, B (MMODA,B)

The LS56 shall receive a buffered, two-bit positive-sense control input for the purpose of defining the input/output protocol, timing and format for all exchanges of input branch data and output decoded data. The MMODA and MMODB control inputs shall be continuously present during all encoding or decoding operation, and may not possess changes of any dynamic meaning. The MMODA,B group inputs meet Type A standards (section 6.1). MMODA,B are nominally static inputs.

The purpose of the MMODA,B control input group is the definition of input/output protocol, timing and format as well as the number of ambiguity states in the synchronization logic (the latter in non-block modes only). The MMODA,B group controls both encoding and decoding I/O. The MMODA,B group are defined in Table 2-3.

2.2.4 Mode Buffer-Length (MBUFL)

The LS56 shall receive a buffered, positive-sense control signal which defines the active length of the decoding buffer (RAM) memory under control of the LS56. The MBUFL control input shall be continuously present during any defined decoding operation and may not possess level changes of any dynamic meaning. The MBUFL input is not used in encoding modes. The MBUFL control meets Type A standards (section 6.1).

The function of MBUFL is to provide control over the decoding buffer RAM addressing logic. MBUFL may be arbitrary in encode mode. The definition of the two states of MBUFL is given in Table 2-3. In block decoding modes, MBUFL may change from

Table 2-3. LS56 Modulation/Format Mode Controls

MMODB, A	MODULATION OR FORMAT	SYNC AMBIGUITY	MBUFL	BUFFER RAM ADDRESS SIZE
10	BPSK	2 States (ab) (bc)	0	1024
01	QPSK	2 States (ab) ($\bar{b}$a)		
00	Single Clock OQPSK	2 States (a'b) ($\bar{b}$a)	1	4096
11	BLOCK	None	0	128
			1	256

block to block; however, a DRESET must accompany any block wherein MBUFL has assumed a new state from that of the previous block.

2.3 Branch Data Interface Signals

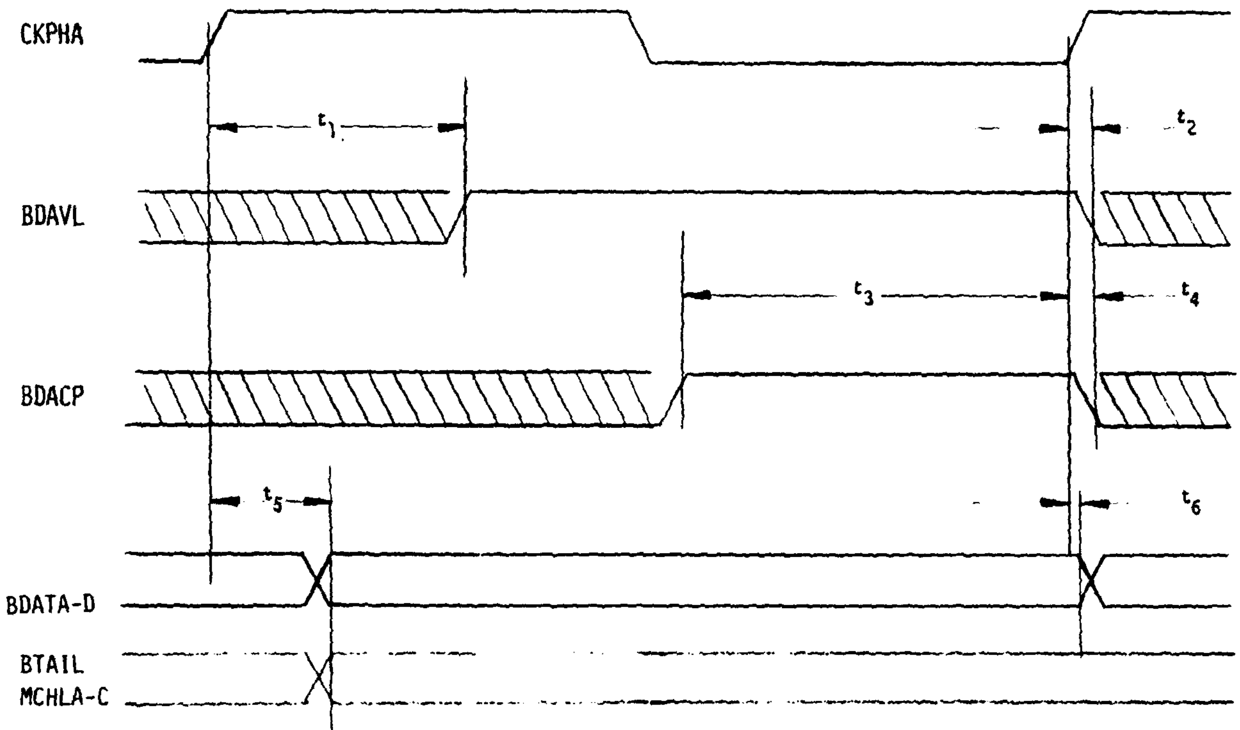
2.3.1 Branch Data Available (BDAVL)

The LS56 shall receive a buffered, positive-sense input signal for the purpose of declaring that the next branch input data is available to the LS56. In block mode, this input BDAVL shall be synchronous with CKPHA. In continuous mode, BDAVL is the demodulator symbol clock. The BDAVL input signal meets Type A standards (section 6.1).

The function of BDAVL is to provide a "data-ready" handshaking signal in block mode, or a "data input required" command signal in continuous mode. In the handshaking mode, BDAVL is potentially reset to its false state with the assertion of the return handshake signal BDACP at the next rising edge of CKPHA. BDAVL may continue to remain true, however, if new branch data is immediately available following a branch data input-accepted cycle of CKPHA. This allows rapid block mode throughput for low-search blocks. BDAVL input timing for one CKPHA clock period is shown in Figure 2-3.

In continuous modes, BDAVL is typically a 50% duty cycle channel symbol rate clock. Under these continuous branch data conditions, the LS56 must service the branch data lines with inputs accepted at least as fast as the symbol rate. Therefore, BDAVL is the highest-priority I/O interrupt of the LS56 decoding behavior. Logic circuitry to synchronize BDAVL with CKPHA then

Figure 2-4. Block Data and Channel Contol Input Timing



TIME	MEANING	MIN	MAX	UNIT
t_1	BDAVL PROPAG DELAY		60	NS
t_2	BDAVL HOLD TIME	5		
t_3	BDACP SETUP TIME	100		
t_4	BDACP HOLD TIME	5		
t_5	BDATA (B,C, ETC) PROPAG DELAY		30	
t_6	BDATA (ETC) HOLD TIME	7		

detect its false-going edge resides in the LS56. Under these conditions, only false-going edges of BDAVL effect input transfers. The BDAVL synchronizing circuit in the LS56 will be designed to assure a sufficiently low probability of metastability on the asynchronous detect for system reliability. Table 2-4 presents a summary of LS56 input/output characturistics for the available decoding modes.

2.3.2 Branch Data Accepted (BDACP)

The LS56 shall provide a *buffered, positive-sense output* signal for the purpose, in block mode, of accepting a branch data input by raising BDACP to logical HIGH when BDAVL is logical HIGH and, in continuous and block decoding modes, an indication that the decoder is caught up and waiting for data. The BDACP output shall be synchronous with CKPHA. BDACP meets Type B standards (section 6.2).

In block mode, the function of BDACP is to accept new branch input data at a rising edge of CKPHA when BDAVL is logical *HIGH for a minimum period of* the CKPHA cycle preceding the transfer edge. BDACP will be in the active (high) state any time the decoder is waiting for input. If the decoder is on a search, BDACP will be in the inactive state. See Figure 2-3 for block mode timing of BDACP.

2.3.3 Branch Data(BDATA,B,C,D)

The LS56 shall receive 4 buffered, positive-sense input signals corresponding to received branch data, interpreted according to the channel defined by MCHLA,B,C. The four inputs BDATA,B,C,D shall be synchronous with CKPHA in block mode, and

synchronous with BDAVL in continuous mode. The four inputs meet Type A standards (section 6.1).

In decoding mode, BDATA,B,C,D inform the LS56 of the demodulator decision at each channel symbol time, regardless of modulation type (QPSK, etc.). These decisions are interpreted according to the channel type in use. The data at BDATA,B,C,D is not decoded immediately, but is transferred to a pipeline register in the LS56 while simultaneously being written into the decoding buffer RAM.

In encoding mode, BDATA alone provides information data to the LS56 for serial encoding at any of 4 code rates: 1/2, 3/4, 7/8 or 1. The protocol for encoding mode transfers at BDATA is identical to that for decoding mode.

Timing of BDATA-D relative to CKPHA and BDAVL (block modes) is given in Figure 2-3.

2.3.4 Branch Tail (BTAIL)

The LS56 shall receive a buffered, positive-sense input signal which (at logical HIGH) identifies the concurrent branch input data as belonging to the encoder-tail portion of a data block. The BTAIL input is identical in signal characteristics to the branch data signals (BDATA, etc.). BTAIL is used exclusively in block mode encoding and decoding; BTAIL must be held at logical LOW for all continuous mode operations.

In decoding, BTAIL informs the LS56 to adopt the interpretation of branch input data as belonging to an encoded-block "tail" wherein all pre-encoded data is known to be zero.

Table 2-4. Input/Output Data Protocol and Timing

DPLOW,	DPURG	DECODER REACTION
0	0	X
0	1	THRESHOLD VIOLATE. CHANGE DECODING DIRECTION
1	0	THRESHOLD SATISFIED. CONTINUE IN SAME DIRECTION, FORWARD OR BACKWARD.
1	1	THRESHOLD SIDEWAYS SATISFIED. MOVE SIDEWAYS.

The complete tail of the encoding process is not transmitted through the channel (only the code parity symbols need be transmitted), and the branch data input sequence consists of pairs of received parity bits only, all of which reflect the existing channel error rate. The LS56, when informed of this unique tail format through the assertion of BTAIL, compensates for the "missing" (i.e., untransmitted) information zeros in such a way that only two computation cycles need be performed for every pair of tail-generated parity bits when decoding these special tail branches. BTAIL is asserted for each pair of tail-parity symbols received through the channel as well as one addition pair of tail-parity symbols appended to the other pairs in order to satisfy packet-environment data protocol.

In encoding, BTAIL acts to force the production of tail parity bits alone. BTAIL is asserted by the encoder interface for a period of CKPHA cycles exactly equal to the total number of tail parity bits desired. It is noted that pairs of tail parity bits appear at the block mode output interface just as pairs of encoded block-proper symbols do when BTAIL is not asserted. When BTAIL is asserted in encoding, the LS56 automatically encodes with logical ZERO data and ignores the BDATA input.

2.4 Decoded Data Interface Signals

2.4.1 Decoded Data Enable (DDENB)

In block mode, the LS56 shall provide a buffered, negative-sense output signal for the purpose of identifying rising edges of CKPHA at which output data shall be synchronously captured by external logic circuitry using CKPHA. This DDENB output shall be synchronous with CKPHA and meet Type B standards (section 6.2).

This shall be true for both block mode decoding and encoding.

In continuous mode, the LS56 shall provide a buffered, postive sense output clock for the purpose of synchronizing encoded or decoded output data. This DDENB output clock may have one of several sources, any of which correspond to the clock applied to the output port of the continuous mode rate buffer. DDENB again meets Type B standards (section 6.2).

In block mode, DDENB allows encoded or decoded data output transactions synchronous with CKPHA. The DDENB is a command enable; the external circuitry must capture pairs of output bits at the edges of CKPHA marked by DDENB since there is no excess output buffering on the LS56. The timing for the DDENB output referenced to a CKPHA period is shown in Figure 2-4.

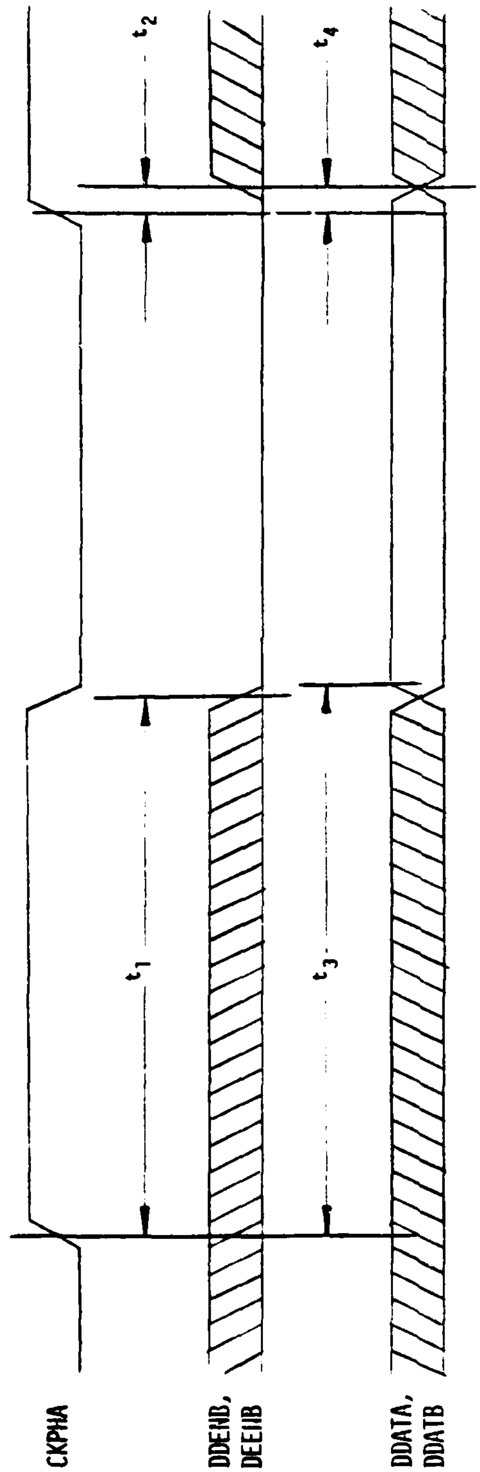
In continuous data operation, DDENB simply provides an output for the particular clock selected to drive the output port of the continuous rate buffer, whether that clock is internally or externally generated.

2.4.2 Decoded Data (DDATA,B)

In all block modes and in all QPSK encoding modes, the LS56 shall provide 2 buffered, positive-sense output signals representing encoded or decoded output data. These outputs DDATA and DDATB shall have synchronism and signal definition in accordance with Table 2-4 and shall meet Type B standards (section 6.2). Timing for DDATA and DDATB is given in Figures 2-4 and K21.

In continuous mode, DDATA,B are synchronous outputs in the

Figure 2-6. Block Mode Data Output Timing



Time	Meaning	Min	Max	Unit
t_1	Enable prop delay		100	ns
t_2	Enable hold time	10		
t_3	Decoded data prop delay		100	
t_4	Decoded data hold time	10		

bit clock for decoding and in the baud clock for encoding (see Table 2-4). For continuous mode decoding and BPSK encoding, the data is output serially on DDATA, with DDATB unused. QPSK encoding utilizes a two-bit parallel output on DDATA and DDATB. In all block decoding and encoding modes, both DDATA and DDATB outputs are active and synchronous with CKPHA.

2.4.3 Detected Error Enable (DEENB)

The LS56 shall provide a buffered, negative-sense output signal for the purpose of enabling an error counter in external circuitry. In block mode, the DEENB output shall be synchronous with CKPHA . In non-block modes, the DEENB output shall be synchronous with the continuous output clock (see Figure K21). The DEENB output shall meet Type B standards (section 6.2).

The function of the DEENB enable is to allow the counting of corrections applied to input branches by the decoder. This count over a fixed number of channel symbols will allow an approximate determination of channel error rate. In block mode, the DEENB enable is the logical OR of the errors detected on a single coded branch. This enable output is a command enable, identical in timing and signal characteristics to DDENB. In continuous mode, DDENB is suitable for use as a clock for a counter, each rising edge indicating the logical OR of four channel errors received on input branches.

The error rate indicated by DEENB at an external counter reflects the actual channel error rate (less a small error due to double-error branches) in block mode decoding, whereas the DEENB output rate reflects the channel error rate divided by 4 in continuous mode decoding (again, less a small error).

2.4.4 Decoder Punctured Clock Output (DPUNC)

The LS56 shall receive a buffered, positive-sense DPUNC control input which defines in BPSK rate 3/4 or 7/8 decoding whether an internally-generated output clock at information rate R or an externally-generated clock at rate R shall be selected by the LS56 for output clocking of the decoder output rate buffer. A DPUNC control input at logical HIGH indicates selection of the internally-generated "punctured" R clock. The DPUNC control input is meaningful only in continuous BPSK decoding; DPUNC is ignored in all other modes. DPUNC meets Type A standards (section 6.1).

2.4.5 Decoder External Output Clock (DECLK)

The LS56 shall receive a buffered, positive-sense DECLK clock input for the purpose of clocking the decoder output rate buffer in rate 3/4 or 7/8 continuous decoding. The DECLK clock input must be of nominal constant period with a maximum instantaneous jitter deviation over four of its cycles of either 5.0% of the CKPHA period or 20 ns, whichever is smaller. The nominal source of DECLK is a PLL of low loop-bandwidth. These characteristics may be relaxed at low decoded data rates. DECLK meets Type A standards (section 6.1).

2.5 Decoder Status Signals

2.5.1 Decoder Status (DSTAT)

The LS56 shall provide a buffered, negative-sense output signal which signals the external decoder system logic that either an auto-plow (plow-forward without decoding initiated by

on-chip logic) is occurring due to imminent overflow of decoding buffer memory or path metric overflow (continuous modes only), or that a path metric overflow has occurred (block mode). Additionally, in TESTX mode, DSTAT provides a single clock cycle enable every time the I/O counter in the sync section overflows (every 64 I/O cycles). This status output has meaning only in decoding modes and is unused in encoder mode. The DSTAT status output shall be an OR of several conditions synchronous in CKPHA. The duration of DSTAT in CKPHA cycles will be the length of plow in continuous mode and will not go away until a reset is asserted in block mode. DSTAT shall meet Type B signal standards (section 6.2).

The function of DSTAT, in normal decoding, is the signaling to the overall decoder system or terminal (post-decoding) equipment that either a) an auto-plow is occurring in continuous mode and the immediately subsequent bits of output data reflect the uncorrected channel error rate, or b) an arithmetic overflow in block mode has occurred. In block decoding, arithmetic overflow indicates that an abort of the particular recieved block should be initiated by the decoder system. Two independent events may cause auto-plow in continuous decoding: a) arithmetic overflow (as in block mode), and b) imminent overflow of the decoding buffer memory space. The decoder will change synchronization state on every auto-plow if the output variable, DSYNC, in it's active (high) state. With DSYNC in the inactive (low) state, the only act of the auto-plow is to plow the decoder forward, outputting the channel error rate for the duration of the plow. Note: In the case of an auto-plow due to arithmetic overflow, there will in general be some typically small fraction of a buffer length of output bits which will contain spurious

corrections in addition to channel-originated errors. For this reason, the arithmetic section of the LS56 has been designed to minimize the probability of arithmetic overflow to an extremely small value.

2.5.2 Decoder Sync-State Transition (DSYNC)

The LS56 shall provide a buffered, positive-sense output signal for the purpose of flagging the external logic thatthe decoder is searching for the proper sync-state. This output, DSYNC, possesses meaning only in continuous decoding modes. The DSYNC status output shall be a state variable synchronous in CKPHA, and will be active (logical HIGH) as long as the possibility of a sync-state change exists. DSYNC shall meet Type B output signal standards (section 6.2).

The function of DSYNC is to flag either certain specialized input logic such as deinterleavers or decoder system control logic that the continuous mode decoder is in the sync search state, abandoning the previous input channel symbol sync state for another sync state due to an unacceptable rate of buffer overflows, metric overflows or a combination of both. With DSYNC active (logical HIGH), the decoder will change sync-state every time a buffer overflow or path metric overflow occurs. The decoder can move out of the sync search state if there occurs no buffer or path metric overflows within 512 input cycles. With DSYNC in the inactive (LOW) state, the decoder can be assumed to be in the correct decoder sync state. If DSYNC never goes to the inactive (low) state, conditions for decoding are unacceptable under present code parameters. The DSYNC output has no meaning in block mode decoding or any encoding mode.

2.5.3 Decoder-Purge Done (DDONE)

The LS56 shall provide a buffered, positive-sense output signal for the purpose of flagging the external logic of having finished a purge of the decoding buffer memory (as initiated by DPURG, section 2.1.4). Teh DDONE status output shall be synchronous with CKPHA, be one CKPHA cycle in duration (logical HIGH), and meet Type B standards (section 6.2). The DDONE status output is meaningful in block mode only.

The function of DDONE is to flag specialized output interface logic that an externally initiated purge of decoding buffer memory is complete and the auto-reset feature of the purge function is done. The occurance of DDONE is actually a number of computation cycles beyond the completion of block mode outputs. This is because tail branch data is not output but the tail parity branches in memory must be "visited" by the buffer addressing before arriving at the final address of the block (i.e., the address at which DPURG was asserted). The length of delay in computation cycles from the last output to the assertion of DDONE is equal to the number of tail parity pairs at the end of the block.

The timing of DDONE is identical to that of DDENB as described in Figure 2-4. Note: An automatic reset is performed internally with the assertion of DDONE in preparation for input of the next block for decoding.

2.6 Decoding Buffer (RAM) Signals

2.6.1 RAM Addresses (RAMA0-11)

The LS56 shall provide 12 buffered address output signals for the purpose of defining the decoding buffer RAM address on all active LS56 cycles. These 12 outputs, RAMA0-11, shall be synchronous with CKPHA and meet Type C standards (section 6.3). See Section 4.1 for a detailed explaination of RAM timing and interconnection.

The function of the RAMA0-11 outputs is the addressing of the decoding buffer RAM memory. The outputs may be directly connected to the RAM devices in use. Both bytes of RAM (A and B, representing branch storage and decision storage respectively) are addressed identically, cycle for cycle. Explicit timing of RAMA0-11 is given in section 4.2.

2.6.2 RAM (write) Data (RAMDO0-7)

The LS56 shall provide a group of 8 buffered, positive-sense output signals for the purpose of supplying write-data to the 8-bit decoding buffer memory (external RAM). These 8 outputs, RAMDO0-7, shall be valid during the second half of CKPHA period during which RAM-writes occur. Explicit timing requirements of the RAMDO0-7 signals are specified in section 4.2. The RAMDO0-7 signals are used only in decoding; encoding operation does not involve the use external RAM in any way. RAMDO0-7 meet signal Type B standards (section 6.2).

The function of RAMDO0-7 is to provide branch, correction and indicator data resulting from either channel I/O cycles or

decoding-only cycles to the decoding buffer memory. This data is generally divided into Branch data (6 bits) and Correction data (2 bits). The 8-bit word organization of this RAM is detailed in Table 2-5.

2.6.3 RAM (read) Data (RAMDI0-7)

The LS56 shall receive a group of 8 buffered, positive-sense data input signals for the purpose of acquiring the contents of the decoding buffer memory at any particular address. These 8 inputs, RAMDI0-7, shall be valid prior to the falling edge of CKPHA when they are latched on-chip using this clock. Explicit timing requirements of the RAMDI0-7 signals are specified in section 4.2. The RAMD0-7 signals are used only in decoding; encoding operation does not involve the use of external RAM in any way. The RAMD0-7 data inputs meet Type A standards (section 6.1).

The function of RAMDI0-7 is to provide a data path for reading the contents of the decoding buffer memory as previously written into this memory through RAMDO0-7. Reading and writing of RAM memory generally occurs each CKPHA cycle. This data is organized identically to that of RAMDO0-7 (see Table 2-5).

The RAM data input/output is organized for a separate I/O memory but with certain external logic, the LS56 can interface with common I/O memory. This external logic is detailed in section 4.2.1.1. High speed RAM data transactions will require independent RAM input and output data paths due to internal delay paths in the LS56.

Table 2-5. Decoding Buffer Memory Organization

FORMAT	MODULATION	CODE RATE	SUPPORTING SYSTEM FEATURES AND RQMTS	I/O CHARACTERISTICS: INPUT DATA (BDATA,B,C,D): PROTOCOL	EXT. CONTROL	INT. CONTROL	DATA TIMING	$\mu > 1$ RQMT	OUTPUT DATA (DDATA,B): PROTOCOL	EXT. CONTROL	INT. CONTROL	OUTPUT DATA	DETECTED ERROR
BLOCK	ANY TYPE	ALL R_C	DMA INTERFACE (IC) OR A PRE-DECODER CHANNEL-BUFFER INTERFACE	HANDSHAKE ALL CASES 2 CODE SYMBOLS PER TRANSACTION (DECODE) 1 BIT PER (ENCODE	BDAVL SYNC IN ϕ CLOCK	BDACP SYNC IN ϕ CLOCK	SYNC IN ϕ CLOCK	$\frac{RR_C^{-1}}{2} < \phi$	IN SYNC WITH OUTPUT ENABLE DDENB CONDITIONED ON INPUT DATA AVAILABLE	SLAVE IN ϕ CLOCK	DDENB SYNC IN ϕ CLOCK	DATA PAIRS SYNC IN ϕ CLOCK	DEENB SYNC IN ϕ CLOCK
CONTINUOUS	QPSK	1/2	NO SPECIAL FEATURES							RELOCK WITH SCLK	$\overline{SCLK}$/ FM INPUT @ RBO	SERIAL SYNC IN SCLK	SERIAL SYNC IN SCLK
		3/4, 7/8	ISOCHRONOUS OUTPUT REQD. PLL ÷ RCLK, SELECTABLE R_C	ISOCHRONOUS INPUT CLOCK (SCLK) 1 SYMBOL/EDGE, (DEC) 1 BIT/EDGE, (ENC)	QPSK SYMBOL CLOCK	EDGE DETECT	SYNC IN SCLK	$R_{SCLK} < \frac{\phi}{2}$	ISOCHRONOUS OUTPUT FROM ON-CHIP RATE BUFFER (RBO REFERS TO OUTPUT SIDE OF RATE BUFFER)	PROVIDE RCLK/ TO CHIP	RCVD RCLK @ RBO	SERIAL SYNC IN RCLK	SERIAL SYNC IN RCLK
	BPSK	1/2	NO SPECIAL FEATURES						SAME AS QPSK BUT W/INTNL CLOCK	SLAVE TO RCVD RCLK	$\overline{SCLK}$/2 + GEN,S RCLK TO RBO	SERIAL SYNC IN RCLK	SERIAL SYNC IN RCLK
		3/4, 7/8	ISOCHRONOUS OUTPUT REQD. PLL ÷ RCLK, SELECTABLE R_C						SAME AS QPSK IN ALL RESPECTS	←	←	←	←
		3/4, 7/8	NONREGULAR OUTPUT ALLOWED NO SPECIAL FEATURES	ISOCHRONOUS INPUT CLOCK (SCLK) 1 SYMBOL/EDGE, (DEC) 1 BIT/EDGE, (ENC)	BPSK SYMBOL CLOCK	÷SCLK/2 S/P CONV	SYNC IN SCLK	$R_{SCLK} < \phi$	INTERNALLY-PUNCTURED SYMBOL CLOCK APPLIED TO ON-CHIP RATE BUFFER	SLAVE TO RCVD PCLK	PCLK DERIVED FM SCLK @ RBO	SERIAL SYNC IN PCLK	SERIAL SYNC IN PCLK

2.6.4 Write RAM Byte A (WRAMA)

The LS56 shall provide a buffered, positive-sense output signal for the purpose of enabling writing the branch byte (Byte A) of the decoding buffer RAM memory. This output signal, WRAMA, is a synchronous enable within the CKPHA cycle and meets Type B standards (section 6.2).

The function of WRAMA is to control write operations into the branch byte of the decoding buffer RAM. This output is determined by the type of cycle in effect. Only cycles involving I/O allow WRAMA operations. See Table 2-6 for the occurance of WRAMA versus decoding cycle type. WRAMA has no meaning in encoder mode. WRAMA actually provides a positive-sense enable (logical HIGH) to an external NAND function which is in turn used to gate a RAM Write pulse. In this sense, WRAMA is a control output synchronous in CKPHA.

2.6.5 Write RAM Byte B (WRAMB)

The LS56 shall provide a buffered, positive-sense output signal for the purpose of writing data into the decision byte (Byte B) of the decoding buffer RAM memory. This output , WRAMB, is a synchronous enable within the CKPHA cycle and meets Type B standards (section 6.2).

The function of WRAMB is to control write operations into the decision byte of the decoding buffer RAM. This output is determined by the type of cycle in effect and the occurance of the logical low sense of CKPHA. Write operations per WRAMB always occur during I/O cycles. Write operations per WRAMB may occur during search cycles, depending upon the direction of

Table 2-6. Buffer Operations

		Block	QPSK	BPSK
Branch Data RAM (A)	$\overline{\text{RAMD0}}$	SGN1(N)	SGNI(N)	SGN(N)
	$\overline{\text{RAMD1}}$	A/M1(N)	MAGI(N)	MAG(N)
	$\overline{\text{RAMD2}}$	SGN2(N)	SGNQ(N)	SGN(N+1)
	$\overline{\text{RAMD3}}$	A/M2(N)	MAGQ(N)	MAG(N+1)
Decision RAM (B)	$\overline{\text{RAMD4}}$	TAIL(N)	SGNI(N-1)	SNG(N+2)
	$\overline{\text{RAMD5}}$	BIND(N)	MAGI(N-1)	MAG(N+2)
	$\overline{\text{RAMD6}}$	C1LST	C1LST	C1LST
	$\overline{\text{RAMD7}}$	C1LST	C2LST	C2LST

search. See Table 2-6 for the occurance of WRAMB versus decoding cycle type. WRAMB has no meaning in encoder mode. WRAMB actually provides a positive-sense enable to an external NAND function which is in turn used to gate a RAM Write pulse. In this sense, WRAMB is a control output synchronous in CKPHA.

2.7 LS56 Chip-Level Test Control

The LS56 logic shall have test features supporting both engineering and production test. The approach used allows external control over decoding cycle results and certain global conditions, such as the synchronization state. Testing of interface and RAM memory control/pipeline logic is possible using simple observation of LS56 input/output and RAM data/address I/O pins (pads). Testing of the encoding function is strictly an output observation problem. Specialized decoder test control is provided by the TESTX control signal.

2.7.1 Test Control Input (TESTX)

The LS56 shall receive a buffered, positive-sense test control generated synchronously with CKPHA by external testing logic. The TESTX input is a global control at the top of the control hierarchy (mode controls being next down in the hierarchy). The CKPHA-synchronous requirement on TESTX shall not hold when any change in TESTX is followed by a global reset (via DRESET). TESTX meets Type A standards (section 6.1).

The function of TESTX is to define a unique control condition useful in testing the decoding function wherein the normal interpretation of input data (e.g., BDATA) and control (e.g., DPLOW, DPURG) is supplanted by new interpretations which

allow virtually immediate control over the "results" of decoding cycles, cycle by cycle. The TESTX control further exercises special control over the syncronization processor to allow rapid testing of the 6-bit and 10-bit counters. TESTX is further useful in providing an unusual pathway for overriding the continuous data mode synchronization process on-chip by an external sync processor in special applications.

2.7.2 Path Metric Counter Not Empty (PMCNE)

The LS56 will output a buffered, positive-sense output, PMCNE, for the purpose in test mode of determining the state of the path-metric counter. PMCNE will be in the active (logical HIGH) state any time the path metric extension counter has one bit set. This is useful, in test mode, for testing the path metric counter and the arithmetic section of the LS56. PMCNE will be active and valid during all decoding and is not used during encoding. PMCNE is synchronous in CKPHA an conforms to type B (section 6.2) signal standards.

2.7.3 Syndrome (S)

The LS56 will provide a buffered, positive-sense output, S, for the purpose in test mode of determining the state of syndrome from the syndrome decoder. This output is useful in test mode to determine that the input section and the syndrome decoder logic is working. S will be in the active (logical HIGH) state any time the syndrome out of the syndrome decoder is one. S will be active during normal decoding and is valid while the decoder is on a single-bit branch while moving forward. S is a synchronous output in CKPHA and conforms to type B (section 6.2) signal standards.

3. ORGANIZATION

3.1 Top Level Architecture

The LS56 can be conceptually partitioned into five basic subfunctions:

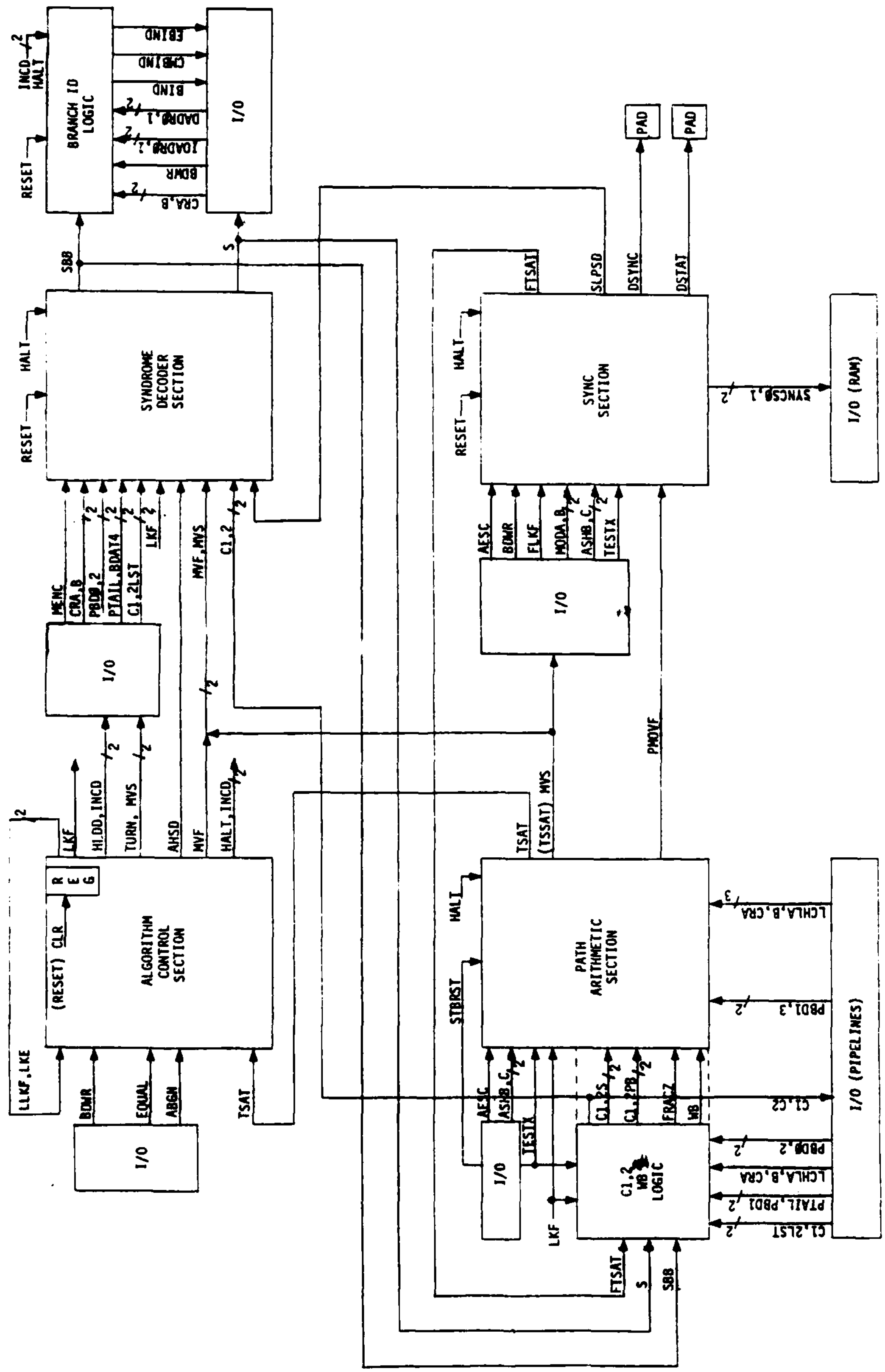
1. Input/Output or I/O
2. Algorithm/Control State Sequencer
3. Syndrome Decoder
4. Path Arithmetic Processor
5. Syncronization Processor

The four non-I/O subfunctions interconnect densely with the I/O, but otherwise only sparsely between themselves. This organization is apparent in Figure 3-1. It is pointed out that these five subfunctions each include peripheral logic which in two cases is substantial enough to warrant identification in Figure 3-1. The Branch identification (BRID) logic is closely associated with the Syndrome Decoder but interconnects with the I/O in a substantially different way than the balance of the Syndrome Decoder subfunction. The C1,2/WB Logic (correction logic) is an autonomous preprocessor required at the front end of the Path Arithmetic subfunction.

The various LS56 modes utilize a limited number of these subfunctions. This is summarized below:

Subfunction	Continuous Data Encoding	Continuous Data Decoding	Block Data Encoding	Block Data Decoding
I/O	X	X	X	X
Algorithm Control	X	X	X	X
Syndrome Decoder	X	X	X	X
Path Arithmetic		X		X
Sync		X		

Figure 3-1. Top Level Architecture



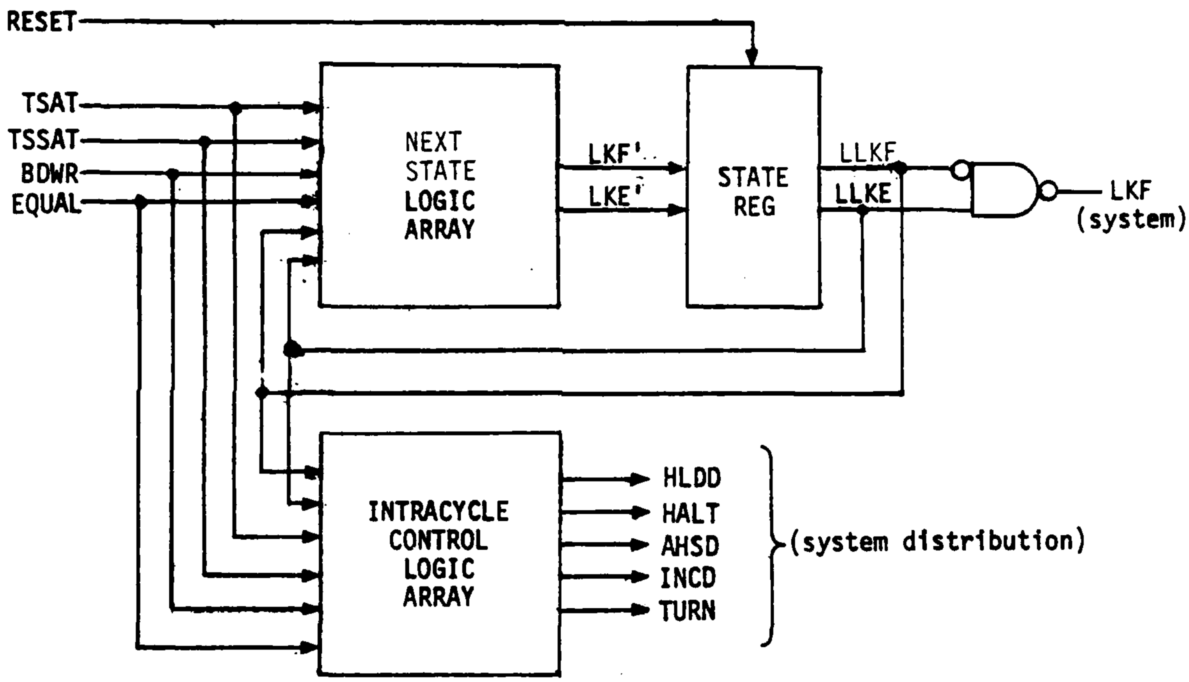
The following sections will discuss the subfunctions in more detail.

3.2 Algorithm/Control State Sequencer

The main control logic of the LS56 is a 4-input, 7-output, 2-bit-state-variable sequencer which controls both the input/output process and the execution of the decoding algorithm. Although the control of these two facets of the LS56 is fully "merged", there are in essence two independent processes: incrementing of the buffer RAM address during I/O, and the LS56 control logic is responsible for directing decoding algorithm actions; decoding address pointer movement, data/correction pipeline movement, syndrome decoder shifting/modifiying and arithmetic register updating. The control pertaining to data input to the LS56 resides in special interrupt logic for continuous data and in a combination of handshake and buffer address status logic for block data, as opposed to being explicitly controlled from the algorithm/control subfunction. A major advantage of this approach is that a single state transition sequence can be defined for all types of I/O case. These state transitions are realized in a classical state machine depicted in Figure 3-2. The state machine executes transitions in accordance with the state diagram of Figure 3-3.

In Figures 3-2 and 3-3, the state variables LLKF and LKE code the activity states of the LS56. A special function of the state variables, LKF, describes the basic encoder or decoder posture of tentatively proceeding forward in the processing of new input data. In encoding, there is only forward processing, while the non-realtime searching of decoding involves forward and backward steps. Thus, the following states are relevant to each

Figure 3-2. Algorithm/Control State Sequencer



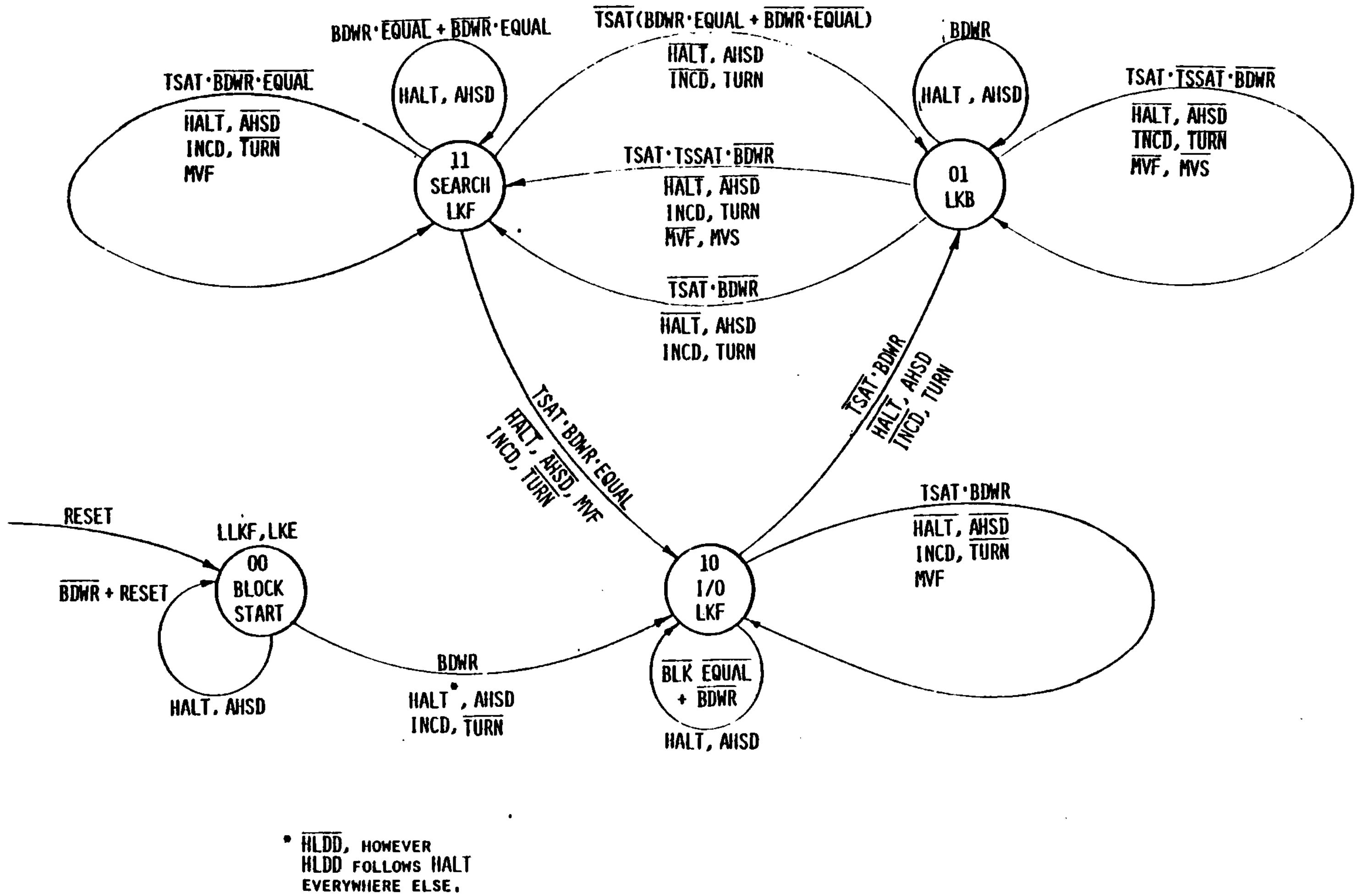


Figure 3.3. Algorithm/Control State Diagram

of the basic LS56 modes:

- Block Encoding: BLOCK START and I/O LKF
- Block Decoding: all four states
- Continuous Encoding: I/O LKF only
- Continuous Decoding: all states except BLOCK START.

State Transitions in Figure 3-3 are characterized by the required input variable values on top of each transition arrow and the resultant output, or control variables on the bottom of each arrow. The precedence of inputs to the state machine is BDWR, EQUAL, TSAT, then TSSAT. This precedence determines the state transition equations listed in Table 3-1. The BDWR variable (Branch Data Write) is the above referenced input derived from an I/O interrupt (continuous data) or a combination of I/O handshake and state conditions (block data). The EQUAL variable is the comparison of the two address pointers (see Section 3.3). The TSAT and TSSAT variables are results of a decoding step and are irrelevant in encoding operation.

The output or control variables of the state machine are widely distributed throughout the LS56. The HLDD and HALT controls are used as synchronous clock enables on most flipflops in the design. The AHSD control is a special synchronous enable used exclusively in the syndrome decoder. The INCD control determines the direction of count for the decoding address pointer (see section 3.3) as well as the branch indicator counter. The TURN output provides directional control to the data and correction pipelines when decoder searches reverse in direction. The MVF and MVS outputs control the sense of syndrome decoder modifications (e.g., shift forward, backward or stand-in-

LS56 DECODER ALGORITHM/CONTROL EXPRESSIONS

$$\begin{aligned}
\mathrm{LLKF'} &= \mathrm{LLKF}\cdot\mathrm{TSAT} + \mathrm{LLKF}\cdot\mathrm{BDWR}\cdot\overline{\mathrm{EQUAL}} + \mathrm{LLKF}\cdot\overline{\mathrm{BDWR}}\cdot\mathrm{EQUAL} \\
&+ \overline{\mathrm{LLKF}}\cdot\mathrm{LKE}\cdot\overline{\mathrm{BDWR}}\cdot\overline{\mathrm{TSAT}} + \overline{\mathrm{LLKF}}\cdot\overline{\mathrm{LKE}}\cdot\mathrm{BDWR} \\
&+ \mathrm{LKE}\cdot\overline{\mathrm{BDWR}}\cdot\mathrm{TSSAT} + \overline{\mathrm{LKE}}\cdot\overline{\mathrm{BDWR}}\cdot\overline{\mathrm{EQUAL}}
\end{aligned}$$

$$\begin{aligned}
\mathrm{LKE'} &= \mathrm{LLKF}\cdot\mathrm{BDWR}\cdot\overline{\mathrm{TSAT}} + \overline{\mathrm{LKFF}}\cdot\mathrm{LKE} \\
&+ \mathrm{LKE}\cdot\overline{\mathrm{TSAT}} + \mathrm{LKE}\cdot\overline{\mathrm{BDWR}} + \mathrm{LKE}\cdot\overline{\mathrm{EQUAL}} + \overline{\mathrm{BLK}}\cdot\overline{\mathrm{EQUAL}}
\end{aligned}$$

NEXT STATE VARIABLES

$$\begin{aligned}
\mathrm{HLDD} &= \mathrm{LLKF}\cdot\overline{\mathrm{BDWR}}\cdot\mathrm{EQUAL} + \overline{\mathrm{LLKF}}\cdot\mathrm{LKE}\cdot\mathrm{BDWR} \\
&+ \overline{\mathrm{LKE}}\cdot\overline{\mathrm{BDWR}} + \mathrm{LKE}\cdot\mathrm{BDWR}\cdot\overline{\mathrm{EQUAL}}
\end{aligned}$$

$$\begin{aligned}
\mathrm{HALT} &= \mathrm{LLKF}\cdot\overline{\mathrm{BDWR}}\cdot\mathrm{EQUAL} + \overline{\mathrm{LLKF}}\cdot\mathrm{BDWR} \\
&+ \overline{\mathrm{LKE}}\cdot\overline{\mathrm{BDWR}} + \mathrm{LKE}\cdot\mathrm{BDWR}\cdot\overline{\mathrm{EQUAL}}
\end{aligned}$$

$$\begin{aligned}
\mathrm{AHSD} &= \overline{\mathrm{LLKF}}\cdot\overline{\mathrm{LKE}} + \overline{\mathrm{LLKF}}\cdot\mathrm{LKE}\cdot\mathrm{BDWR} + \overline{\mathrm{TSAT}} \\
&+ \mathrm{LLKF}\cdot\overline{\mathrm{BDWR}}\cdot\mathrm{EQUAL} + \mathrm{LKE}\cdot\mathrm{BDWR}\cdot\overline{\mathrm{EQUAL}} + \overline{\mathrm{LKE}}\cdot\overline{\mathrm{BDWR}}
\end{aligned}$$

$$\begin{aligned}
\mathrm{INCD} &= \mathrm{LLKF}\cdot\mathrm{TSAT} + \overline{\mathrm{LLKF}}\cdot\overline{\mathrm{TSAT}} + \overline{\mathrm{LLKF}}\cdot\mathrm{BDWR} + \mathrm{TSSAT} \\
\mathrm{TURN} &= \mathrm{LLKF}\cdot\overline{\mathrm{TSAT}} + \mathrm{LKE}\cdot\overline{\mathrm{TSAT}} + \mathrm{LKE}\cdot\mathrm{TSSAT} \\
\mathrm{MVF} &= \mathrm{LLKF}\cdot\mathrm{TSAT} \\
\mathrm{MVS} &= \mathrm{TSSAT}
\end{aligned}$$

INTRACYCLE CONTROL VARIABLES

Table 3.1. LS56 Decoder Algorithm/Control Expressions

place parity alter).

The state decode LKF assures that only the LKB (Look Backward) state is signalled by LKF=0. This is important in block mode wherein the BLOCK START state is actually a "forward" or LKF state.

3.3 Data Input, RAM I/O and Pipelines

The LS56 data input can occur in one of three basic ways: block, BPSK and QPSK. The block case is distinguished by its handshake protocol through BDAVL and BDACP, while the BPSK and QPSK continuous data cases utilize an input clock at BDAVL. In the continuous data cases, the input BDATA-D are received in their synchronous clock (BDAVL) then either directly forwarded (QPSK) or serial/parallel converted (BPSK). This organization is reflected in Figure 3-4.

Once received, the reconfigured data is written into memory during an input (BDWR) cycle. The decoder has the option of selecting the input data or data frommemory for input to it's data pipeline. In continuous mode decoding, the data selected into the pipeline is further manipulated to account for the synchronization state of the decoder. This is accomplished by the multiplexor selected by SYNCS0,1 in Figure 3-4.

The two pipelines of Figure 3-4 act to "extend" the RAM memory into the LS56. This is needed because of delays in reading old and new branch and correction data from RAM. Branch data is pre-fetched the cycle before it is to be used by the decoder. This requires the branch data pipeline to be two stages in length to support decoder search direction changes. The decoder

LS56 DECODER ALGORITHM/CONTROL EXPRESSIONS

$$\begin{aligned}
\mathrm{LLKF'} &= \mathrm{LLKF}\cdot\mathrm{TSAT} + \mathrm{LLKF}\cdot\mathrm{BDWR}\cdot\overline{\mathrm{EQUAL}} + \mathrm{LLKF}\cdot\overline{\mathrm{BDWR}}\cdot\mathrm{EQUAL} \\
&\quad + \overline{\mathrm{LLKF}}\cdot\mathrm{LKE}\cdot\overline{\mathrm{BDWR}}\cdot\overline{\mathrm{TSAT}} + \overline{\mathrm{LLKF}}\cdot\overline{\mathrm{LKE}}\cdot\mathrm{BDWR} \\
&\quad + \mathrm{LKE}\cdot\overline{\mathrm{BDWR}}\cdot\mathrm{TSSAT} + \overline{\mathrm{LKE}}\cdot\overline{\mathrm{BDWR}}\cdot\overline{\mathrm{EQUAL}}
\end{aligned}$$

$$\begin{aligned}
\mathrm{LKE'} &= \mathrm{LLKF}\cdot\mathrm{BDWR}\cdot\overline{\mathrm{TSAT}} + \overline{\mathrm{LKFF}}\cdot\mathrm{LKE} \\
&\quad + \mathrm{LKE}\cdot\overline{\mathrm{TSAT}} + \mathrm{LKE}\cdot\overline{\mathrm{BDWR}} + \mathrm{LKE}\cdot\overline{\mathrm{EQUAL}} + \overline{\mathrm{BLK}}\cdot\overline{\mathrm{EQUAL}}
\end{aligned}$$

NEXT STATE VARIABLES

$$\begin{aligned}
\mathrm{HLDD} &= \mathrm{LLKF}\cdot\overline{\mathrm{BDWR}}\cdot\mathrm{EQUAL} + \overline{\mathrm{LLKF}}\cdot\mathrm{LKE}\cdot\mathrm{BDWR} \\
&\quad + \overline{\mathrm{LKE}}\cdot\overline{\mathrm{BDWR}} + \mathrm{LKE}\cdot\mathrm{BDWR}\cdot\overline{\mathrm{EQUAL}}
\end{aligned}$$

$$\begin{aligned}
\mathrm{HALT} &= \mathrm{LLKF}\cdot\overline{\mathrm{BDWR}}\cdot\mathrm{EQUAL} + \overline{\mathrm{LLKF}}\cdot\mathrm{BDWR} \\
&\quad + \overline{\mathrm{LKE}}\cdot\overline{\mathrm{BDWR}} + \mathrm{LKE}\cdot\mathrm{BDWR}\cdot\overline{\mathrm{EQUAL}}
\end{aligned}$$

$$\begin{aligned}
\mathrm{AHSD} &= \overline{\mathrm{LLKF}}\cdot\overline{\mathrm{LKE}} + \overline{\mathrm{LLKF}}\cdot\mathrm{LKE}\cdot\mathrm{BDWR} + \overline{\mathrm{TSAT}} \\
&\quad + \mathrm{LLKF}\cdot\overline{\mathrm{BDWR}}\cdot\mathrm{EQUAL} + \mathrm{LKE}\cdot\mathrm{BDWR}\cdot\overline{\mathrm{EQUAL}} + \overline{\mathrm{LKE}}\cdot\overline{\mathrm{BDWR}}
\end{aligned}$$

$$\begin{aligned}
\mathrm{INCD} &= \mathrm{LLKF}\cdot\mathrm{TSAT} + \overline{\mathrm{LLKF}}\cdot\overline{\mathrm{TSAT}} + \overline{\mathrm{LLKF}}\cdot\mathrm{BDWR} + \mathrm{TSSAT} \\
\mathrm{TURN} &= \mathrm{LLKF}\cdot\overline{\mathrm{TSAT}} + \mathrm{LKE}\cdot\overline{\mathrm{TSAT}} + \mathrm{LKE}\cdot\mathrm{TSSAT} \\
\mathrm{MVF} &= \mathrm{LLKF}\cdot\mathrm{TSAT} \\
\mathrm{MVS} &= \mathrm{TSSAT}
\end{aligned}$$

INTRACYCLE CONTROL VARIABLES

Table 3.1. LS56 Decoder Algorithm/Control Expressions

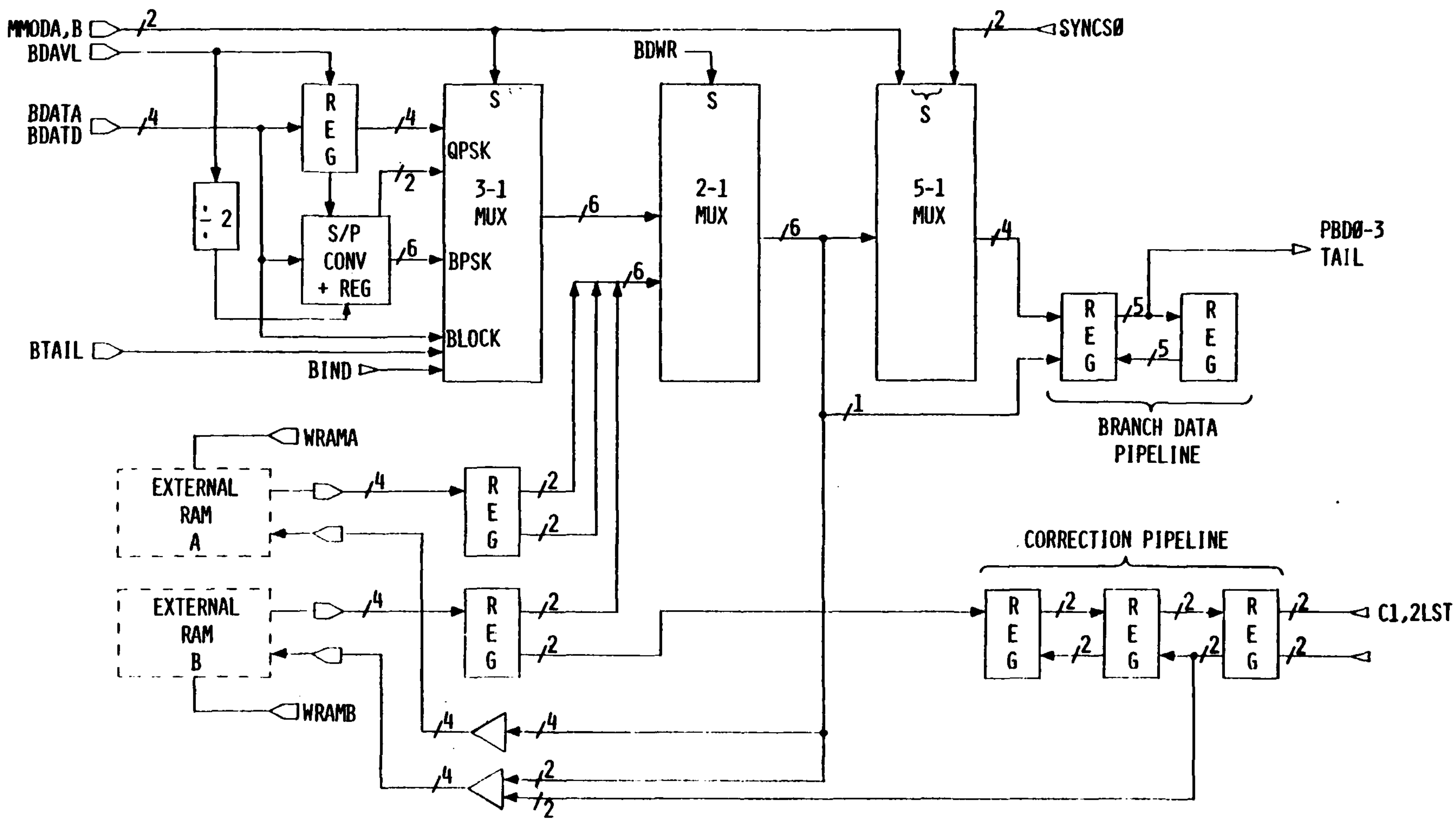


Figure 3-4. Data Input, RAM I/O and Pipelines

corrections are written to RAM in address locations offset by two from the data they apply to. This requires a correction pipeline three stages in length to match corrections up with the data when the decoder searches backward. The output logic takes the offset of corrections from data in RAM into account and matches them up for output.

The external RAM memory can be organized in any of several ways depending upon use of the decoder. All of these ways are, however, canonically identical and appear as two X-word by 4-bit RAM's, RAM A and B, as shown in Figure 3-4. The writing of these RAM's is determined within the LS56 by the WRAMA/B logic. Additional external circuitry is required to correctly time reading and writing operations. As covered in section 3.4, the RAM space may be 128, 256, 1024 or 4096 words in size. The RAM memory is be eliminated for encoder operation.

3.4 RAM Addressing

The LS56 RAM addressing consists of two internal pointers, I/O and decode, that can be selectivly multiplexed to the LS56 RAM address outputs, depending on operating mode and the type of cycle the decoder is operating in. There is also address detect logic that keeps the decoder from accessing forbidden data within the buffer memory. In block mode, the decode pointer is always output from the LS56 since I/O is done only when the decoder is caught up and is waiting for data. In continuous mode,the I/O and decode process are separate unless the decoder is caught up and waiting for data. Therefore, the I/O pointer is output during I/O cycle steals and the decode pointer is output during decode cycles.

The RAM addressing space can be one of four sizes, depending on the decoder operating mode, either continuous or block, and the state of the control variable, MBUFL. In block mode, the address space may either be 256 or 128 with MBUFL 1 or 0 respectivly. The continuous mode buffer size may be either 4096 or 1024 with MBUFL 1 or 0 respectivly. The bits of the RAM address pointers that are not used in smaller buffers than 4096 are reset to zero, allowing use of smaller buffer lengths inside larger external RAM.

The LS6 RAM address outputs aregenerated from twelve latches whose input is a 2-1 mux. This mux selects the output of the I/O pointer or the next the state decode pointer is counting to. The control of the mux is the continuous mode I/O cycle steal detect (STDET). This variable indicates that the next decoding cycle will be an I/O cycle steal. STDET selects the next decoder address in block mode always. This method of outputting the RAM address reduces the delay to having the RAM address valid after the rising edge of the clock. See figure 3-5 for a block diagram of the decode-I/O pointer configuration.

The decode pointer is an up/down counter with the capability of counting up or down by 1 or 3, depending on the movment of the decoder in a search. The decode pointer counts up and down by 1 if the direction of search continues in the same direction, forward or backward, at the end of a decoding cycle. The decode pointer counts up or down by three if the direction of search changes from backward-to-forward or forward-to-backward respectivly. This count by three feature is required due to the data pre-fetch of the decoder. This pre-fetch retrieves from memory the data the decoder will need in the next decoding cycle

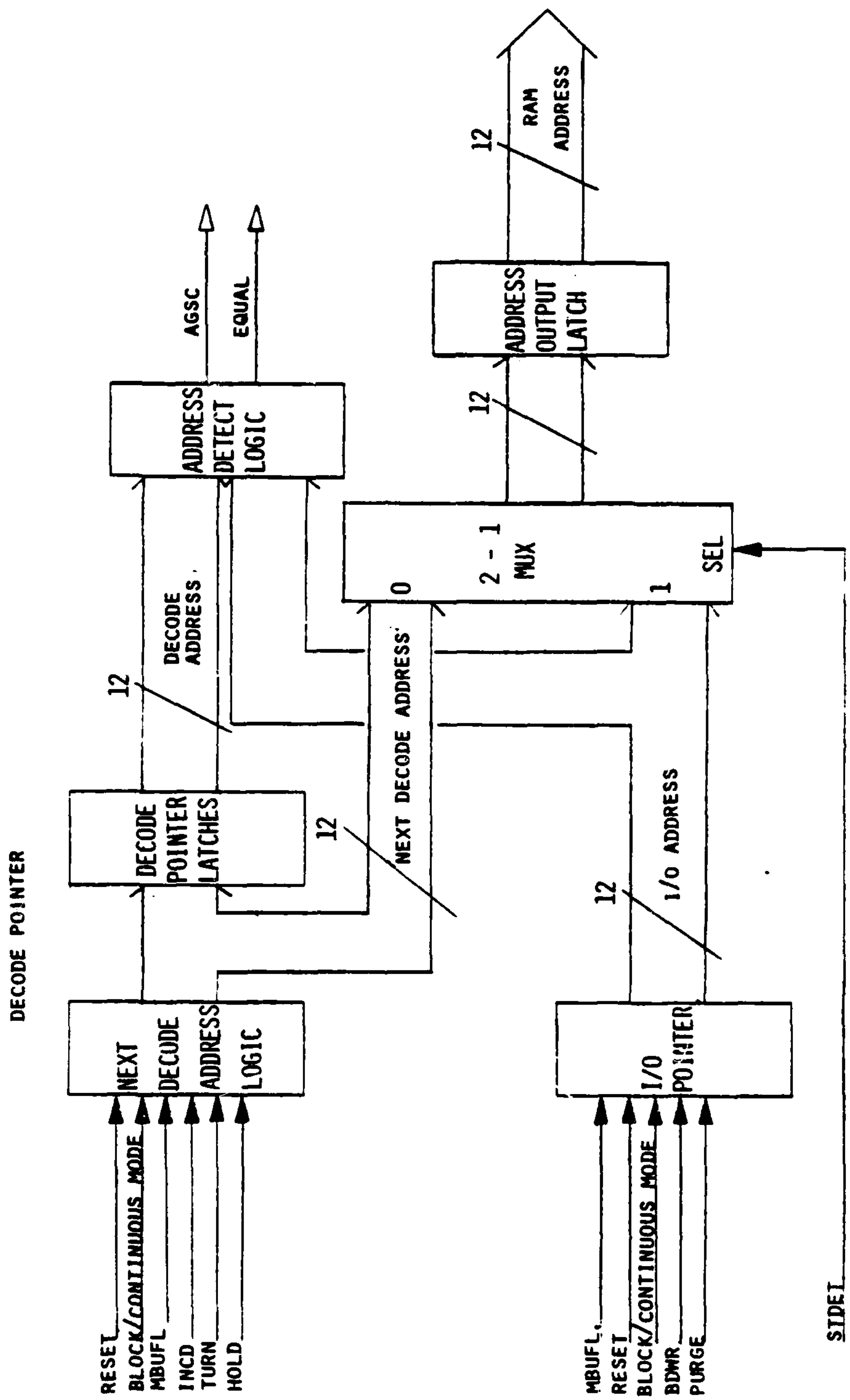


Figure 3-5. Decode - I/O Pointer Configuration

if it continues in the present direction. The decoder retains, in internal pipeline, the data necessary to allow the decoder to turn around. The variables that control the decode pointer are INCD and TURN. INCD follows the direction of decoder search, and is 1 while the decoder moves forward and 0 if the decoder moves backward as the result of the decoding cycle. TURN is the variable that indicates a count by three should be done if it is 1. TURN is asserted any time the decoder switches the search direction. The up or down count by three is controlled by the state of INCD. The output of the decode pointer count logic, prior to the decode pointer latches, is the next decoder address. This goes to the address output mux/latch for the RAM address output.

The I/O pointer is a twelve-bit up counter that counts on I/O cycles. The state of the I/O counter output is the next address that an I/O operation will happen at. This output goes to the address detect logic and the RAM address output mux/latch.

The address detect logic has several types of detects to keep the decoder from accessing forbidden data within the buffer memory. The first is a simple detect that the I/O and decode pointers are equal while the decoder is moving forawrd. This causes the decoder to stop and wait for new data. The remaining address detects keep the decoder from moving backward into data that should not be accessed. These detects include: a simple bump detect that the decoder pointer has moved backward into the I/O pointer, used in block and continuous mode; a zero address detect after reset that keeps the decoder from moving backward past the zero address until the buffer memory is filled, used in block mode only; and a block boundry detect that keeps the

decoder from moving backward into a previously decoded contiguous block. Each of these detects causes the decoder to take imediate action to turn around and go forward. In block mode, the action taken is to loosen the decoder threshold and go forward. The continuous mode action is to turn around on the next decoding cycle and plow forward.

3.5 Branch Timing

The LS56 branch timing is two-bit up/down Johnson counter that keeps track of the type of branch within a branch-group that the decoder is considering. A branch-group is all the data that corresponds to a particular parity bit in the decoder input data. The counter has different modes of operation, depending on code rate and whether the LS56 is used in encode or decode mode.

The representation of each state of the two-bit counter with regards to the type of branch the decoder is considering is as follows:

BRIDB	BRIDA	Type of Branch
0	1	First double-bit branch (DBB) Rate 7/8, 3/4 Rate 1/2, 1, and reset
1	1	Second DBB Rate 7/8
1	0	Third DBB Rate 7/8
0	0	Single-bit branch Rate 7/8 and 3/4

The count sequence follows the top-to-bottom sequence while the decoder is moving forward, and the bottom-to-top sequence moving backward. In rate 7/8, all four states of the counter are used, rate 3/4 only uses the first and last, toggling BRIDA while

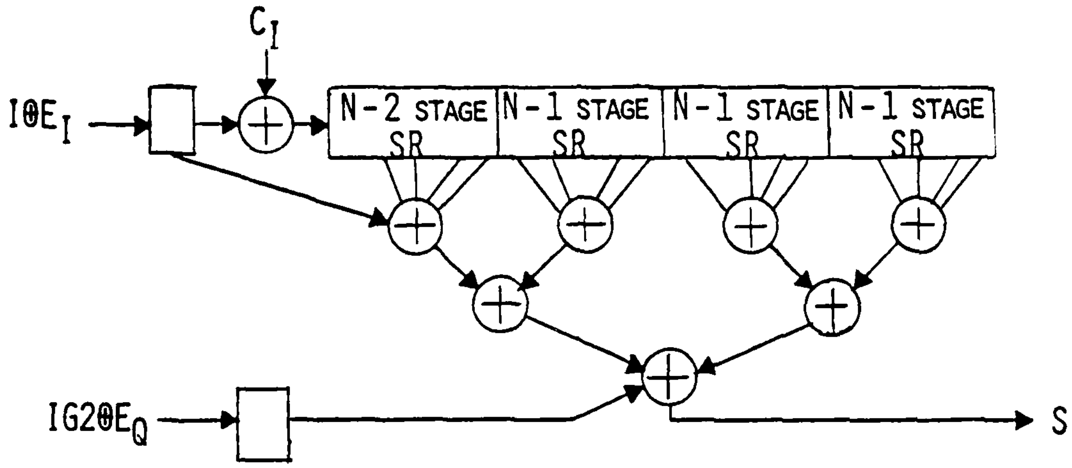
holding BRIDB at zero. The BRID counter is reset to the 01 state during a decoder reset as well as in rate 1/2, rate 1, and tail operation. The BRID counter has, as an up/down control, the variable INCD. INCD is also the variable that controls the up/down count for the decode pointer.

In encode mode, since data is presented to the LS56 serially and is packed into two-bit branches in rate 7/8 and 3/4, the BRID counter must be held in every state but the 00 state to allow two bits of input to the syndrome decoder. The 00 state is the single-bit branch and the parity bit is generated during the cycle it is presented to the syndrome decoder. The hold is accoplished by another counter bit that causes the BRID counter to hold for two data inputs when the BRID counter is in every state except the 00 state.

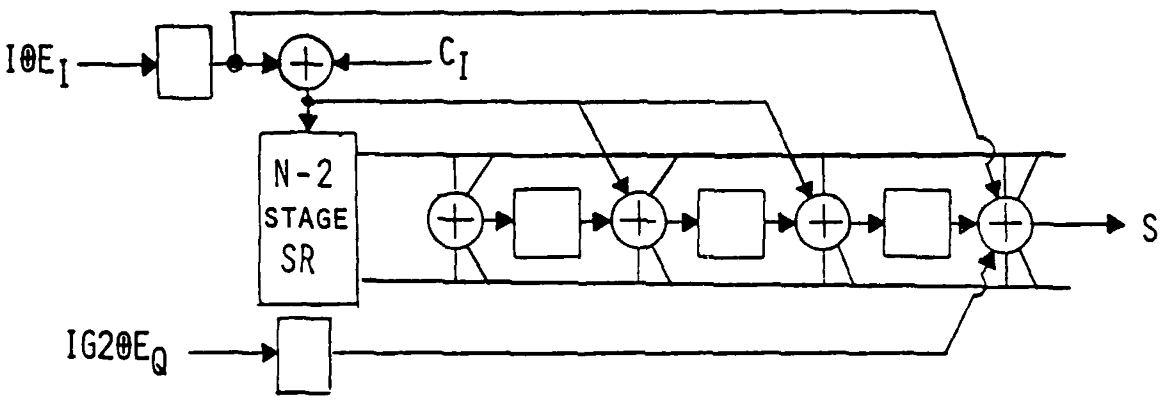
3.6 Syndrome Decoder

The LS56 syndrome decoder operates on the received channel symbols in such a way as to generate a proposed parity bit which is then compared with the received parity bit to generate syndrome. The decoder forces syndrome to be zero by selectivly applying corrections to the received data. The cannonical systematic code syndrome decoder is shown in figure 3-6. It takes the incoming channel symbols (I), adds corrections, and shifts it into a shift register. Selected taps fo the shift register are input to a parity tree, the output of which is the proposed parity bit that is compared to the received parity (IG) to generate syndrome. The type of syndrome decoder implemented in the LS56 is a parity-memory syndrome decoder shown in figure 3-6. It does a series of parital calculations on the input symbols the last of which is the same proposed parity bit

Figure 3-6. Cannonical Systematic Code Syndrome Decoder



CANONICAL SYSTEMATIC-CODE SYNDROME DECODER, $R_C = N\text{-}1/N$



calculated with the shift-register parity tree approach described above. The proposed parity is compared as before with the received parity to generate syndrome.

The syndrome decoder operates on a branch-group of data at a time. In rate 7/8, a branch-group consists of seven bits of data and the corresponding parity bit, a branch-group in rate 3/4 is three bits of data and the parity bit, and in rate 1/2 a branch-group is one data bit and a parity bit. The syndrome decoder collects, in an input register, a branch-group worth of data along with the corresponding corrections, calculates syndrome, and then shifts itself either forward or backward, depending on the decision of the algorithm and control section of the decoder. If the decoder makes a correction on a particular bit of data, the correction is either latched in the input register or the decoder will selectivly invert parity stages, depending on where the decoder is in a search.

The decoder is set up for a constraint length 91 code for rate 7/8, a constraint length 63 code for rate 3/4, and a constraint length 36 code for rate 1/2. Since the decoder operates on branch-groups of data, the actual number of parity stages needed for rate 7/8 is 13, rate 3/4 has 21, and rate 1/2 needs 36 stages. The syndrome calculation stage is the final parity stage for each of the code rates. The parity stages of the syndrome decoder are arranged such that the first 12 stages are used in rate 7/8, 3/4, and 1/2; stages 13 through 20 are used in rates 3/4 and 1/2; and stages 21 through 35 are used in rate 1/2 only. See figure 3-7 for a block diagram of the syndrome decoder. Parity stages that are not used in a particular code rate are reset to a zero state.

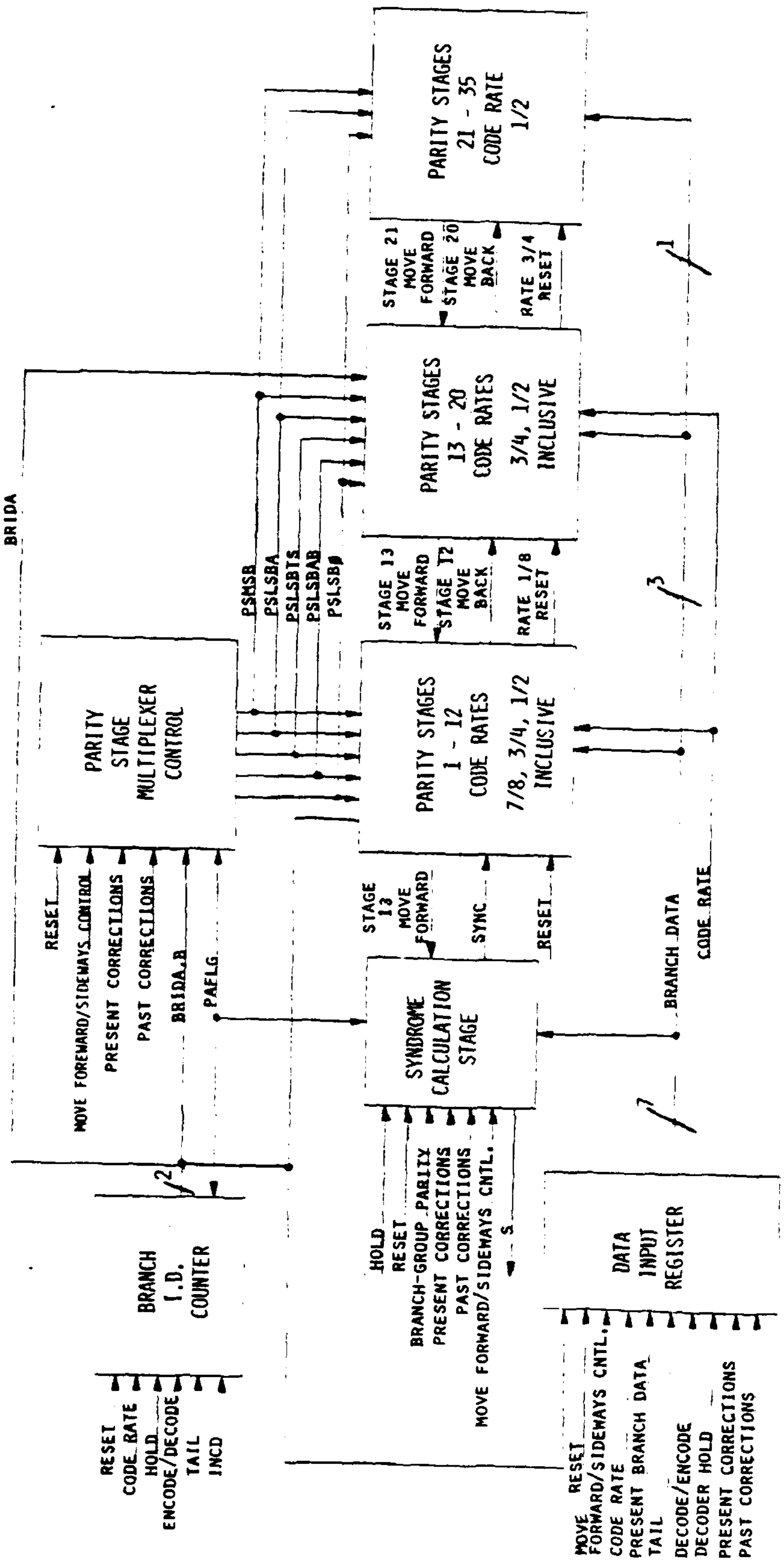


Figure 3-7. Syndrome Decoder Block Diagram

The state of the decoder that indicates how corrections are applied to the decoder and when shifts of the decoder forward or backward should be done is parity alter flag (PAFLG). PAFLG indicates to the decoder that the data that is presently considering has been shifted into the decoder already. This situation arises when the decoder has moved backward out of the branch-group at the furthest point of forward search on a given path in the tree. If the decoder is searching through a branch-group that has not been shifted into the syndrome decoder yet, PAFLG is zero and the corrections are applied to the data at the register. If the decoder is considering data that has been shifted into the decoder, PAFLG is one, and corrections are removed and new corrections are added by selectivly inverting parity stages of the decoder. PAFLG also indicates when the decoder is to do a shift backward or forward. The syndrome decoder shifts forward on a single-bit branch only when PAFLG is 0 and the algorithm and control section says the decoder may move forward. The syndrome decoder shifts backward only on the first branch of the branch-group when PAFLG is 1 and the algorithm/control section says to move backward.

The individual parity stages consist of a latch with a 4-1 multiplexer on it's input. The multiplexer has on it's input the data to hold or invert that stage as well as the data for shifting the decoder forward or backward. See figure 3-8 for a block diagram of the parity stage and how it interconnects to other stages. The data for moving forward or backward comes from a parity tree between the parity stages that has the corrected input data from the input register. The data that is connected to a given stage is dependant on the code and code rate used. A 2-1 multiplexer selects the data from the parity stage one in

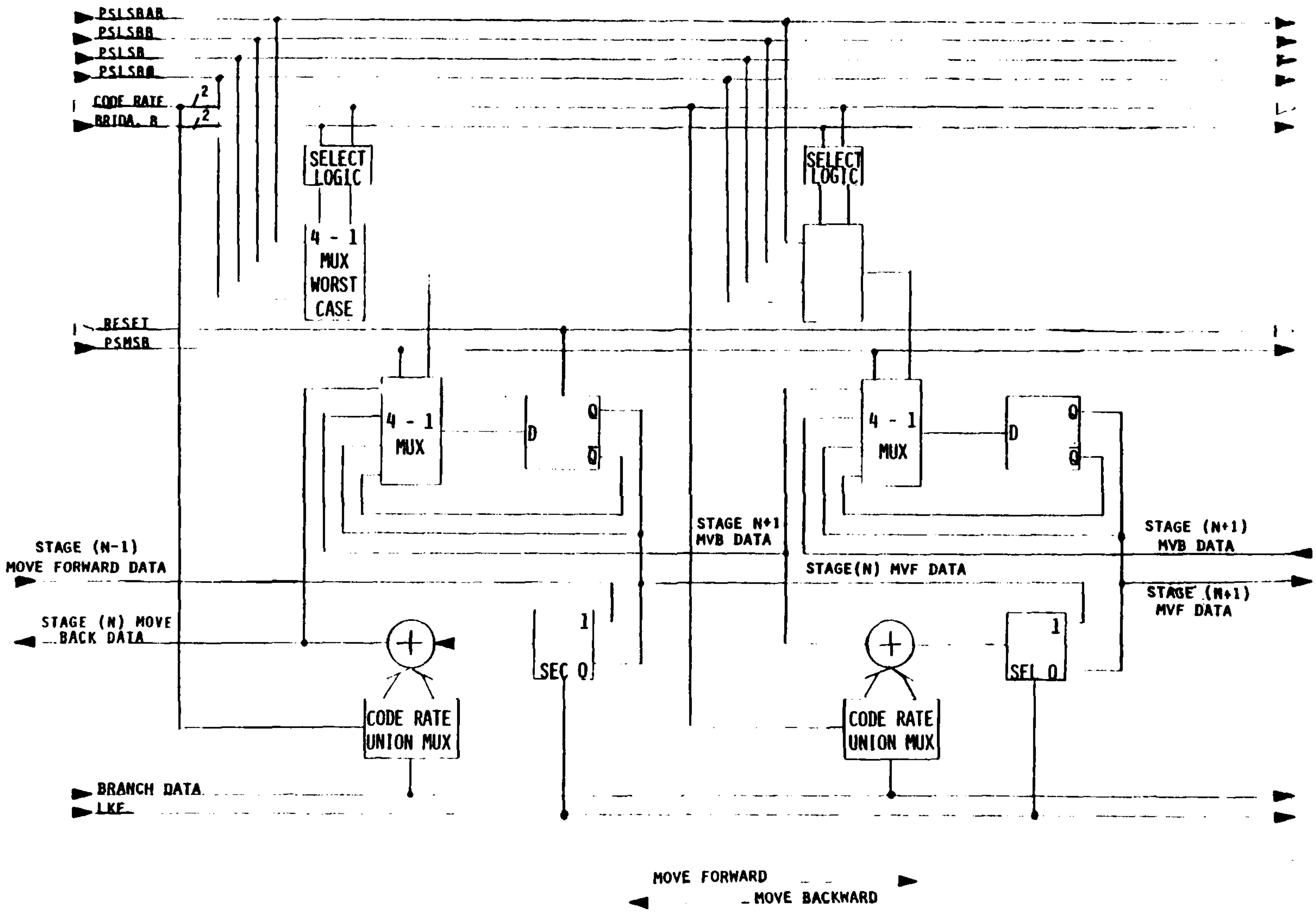


Figure 3-8. Syndrome Decoder Parity Stages

front of or one behind the parity stage in in question as the last input to the parity tree to allow the decoder to shift forward or backward. The select control for the 4-1 mux comes PSMSB and PSLSBA, B, AB, and 0. These control lines are selected for the mux control depending on code rate and the type of branch the decoder is considering. PSMSB is an indication that the decoder is going to do a shift forward or backward. The PSLSB lines control the inversion or holding of a paritcular parity stage while the decoder is not shifting forward or backward and control the direction of shift when the decoder shifts. The PSLSB controls are logically connected to selected parity stages, depending on code rate and branch type. PSLSBA is connected to parity stages where the first bit of the data pair that the decoder is considering is connected to that stage; PSLSBB is used when the second bit of the data pair is connected to a stage; PSLSBAB is used where both bits are connected to a parity stage; and PSLSB0 is used where neither bit is connected to a parity stage. The PSLSB control is zero when PAFLG is zero and PSMSB controls when the decoder shifts forward. When PAFLG is one, old corrections are removed as the decoder moves backward and new corrections are added while the decoder moves forward by selectivly inverting parity stages with the PSLSB control.

The syndrome calculation stage calculates syndrome every time the decoder is going to try to move forward out of a branch-group. There are two methods for calculating syndrome. The first method is used when the decoder is considering data that has not been shifted into the syndrome decoder. Syndrome (S) is calculated by taking parity over all the data in the syndrome input register along with the parity bit associated with the branch-group and the contents of first parity stage of the

syndrome decoder. This is implemented by pre-calculating parity over the double-bit branch data of a branch-group the cycle before S is generated and latching it, generating LPT. LPT goes into a parity calculation with the last data bit and the parity bit of the branch-group and the data in parity stage 1. This method reduces the number of gate delays to generation of S to a minimum. The other method for generation of S is used when the data the decoder is considering has been shifted into the decoder. This method makes use of the fact that S had to be zero when the decoder last moved forward out of the branch-group. It keeps track of parity over old corrections that are removed and new corrections that are added to the branch-group. This parity over corrections is also used to shift the decoder backwards and provide the correct data for the first parity stage. The first and second method for generating S are mux'ed together, the output of the mux being S.

Another function of the syndrome calculation stage is to caulate the correct parity to shift into the first parity stage to allow the syndrome decoder to shift backward. This variable, SYN, is parity over all the data and old corrections for the branch-group the decoder is moving backward out of.

The syndrome decoder input register is responsible for presenting a branch-group of data to the syndrome decoder such that the decoder may move forward or backward on that data. With PAFLG 0, the input register latches data XOR'ed with the corresponding corrections as the data comes in through the present branch data pipelines. The input register latches the uncorrected data with PAFLG 0. The control variables that enable selected latches within the input register are BRIDA,B. This

enabling catches branches as they are input from the PBD registers and holds them until the syndrome decoder may shift forward or backward.

3.7 LS56 Data Output

The LS56 has two output buffers, one for block mode and one for continuous mode. The block mode buffer is a two-bit interface with a negative-sense enable, synchronous with the computation clock, to tell external logic when to take data from the interface. The continuous mode buffer is an asynchronous interface between the decoder clock and an output clock that is phase-locked to the input clock. The output in continuous mode is serial in all decoding modes and BPSK encoding, and is two-bit parrallel in QPSK encoding. Each of the buffers has an output that gives an indication to the number of corrections the decoder is making. The data output for block and continuous modes is done on the same output pins of the LS56.

The block mode buffer takes data out of the buffer memory that is one buffer length old during an I/O cycle and matches it with the corrections the decoder made on that data. The corrected data is then latched in an output buffer that converts the data into two-bit parrallel output. When the buffer has two bits of data to output, a negative-sense enable, DDENB/, will allow external logic to take the data. The corrections are or'ed together and output as a negative-sense enable, DEENB/, for an external counter to count the number of errors the decoder corrected. The corrected data comes out of the LS56 on DDATA,B and may be captured synchronously using DDENB/ as the enable.

The continuous mode output also takes data out of the

buffer memory that is one buffer length old during an I/O cycle and matches it with the corresponding corrections. The corrected data then goes into an asynchronous FIFO that is six bits in depth. The input of the FIFO is clocked in the decoder clock and the output is clocked by a clock that is phase-locked to the decoder input clock. This causes the FIFO input data rate to equal the FIFO output data rate. The data output is through DDATA for serial output and DDATA, B for two-bit parrallel output. The data changes on rising edges of the output clock (OCLK) and will typically be captured by external logic on falling edges of OCLK. The output buffer also provides an output, DEENB/, which and is clocked in the output clock and provides a corrected error indication.

The output clock (OCLK) for the continuous mode output has several different origins, depending on the state of several mode control signals. In decode mode, OCLK is generated in one of four ways. In QPSK rate 1/2, OCLK is the input clock, BDAVL. In QPSK rate 3/4 or 7/8, OCLK is generated from an input to the LS56, DECLK, which is phase-locked to the input clock. In BPSK decoding, rate 1/2 uses the input clock divided by two for OCLK and in RATES 3/4 and 7/8, OCLK may either be generated from the input DECLK, or from a punctured clock based on the input clock that has occasional rising edges removed so that the data output rate is correct but the data output is not regular. The control over whether the punctured clock or the phase-locked DECLK is used for output is exerted by the external control variable, DPUNC. With DPUNC 0, DECLK is used, and with DPUNC 1 the punctured clock is used.

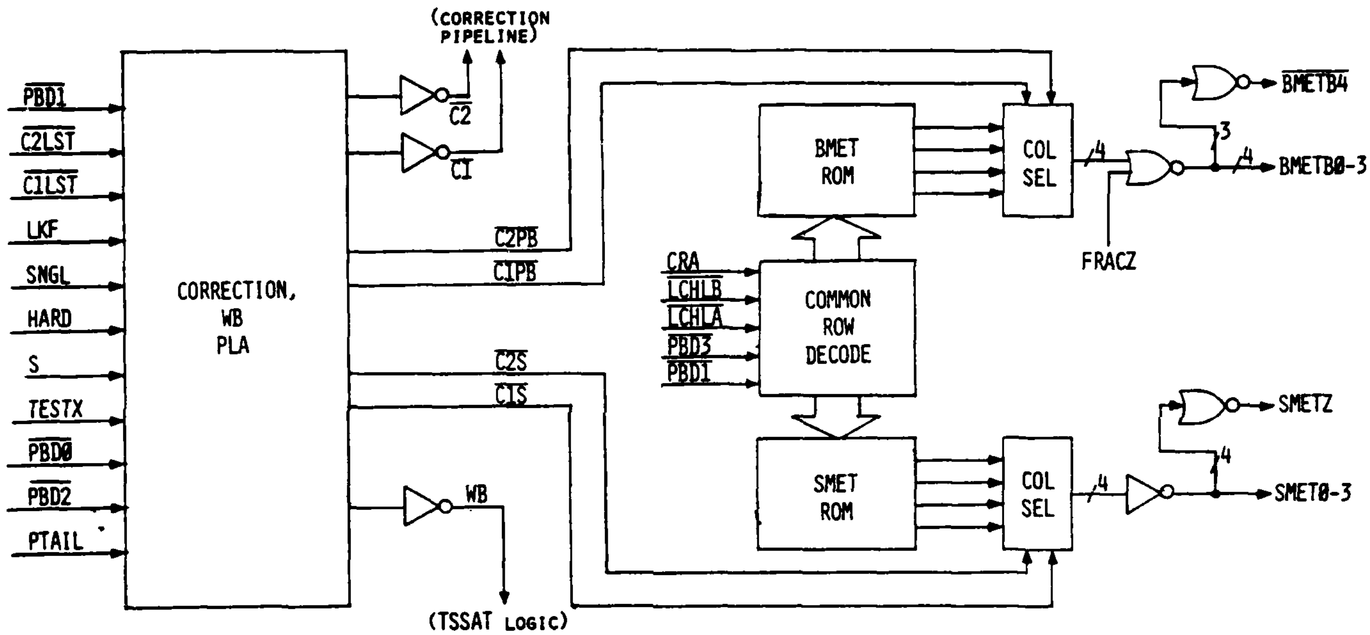
3.8 Correction, WB and Branch Metric Logic

The correction, Worst Branch (WB) and Branch Metric (BMET) logic is the preprocessing required at the input to the Path Arithmetic Processor. This logic is essentially part of the arithmetic processing, but is functionally self-contained. The logic acts as an interpreter of incoming channel data, mode and current state of the syndrome decoder. It reduces these many inputs to a pair of tentative corrections to be eventually stored in the buffer RAM and two 5-bit 2's-complement numbers BMET and SMET which are analyzed in the Path Arithmetic Processor during each decoding cycle. The organization of this logic is shown in Figure 3-9.

The reduction of data takes place from left to right in Figure 3-9. The primary outputs of this logic are C1,C2 (to the correction pipeline), WB (to the TSSAT portion of the Path Arithmetic) and the two 5-bit numbers BMETB (0-4) and SMET (0-3 and Z). SMET is actually a compound number composed of a 4-bit 2's-complement section and a 1-bit zero-value-indicator (SMETZ). Intermediate results in this logic include C1,2PB and C1,2S which are used to determine the BMET and SMET numbers respectively. One data reduction is not shown in Figure 3-9: this is the derivation of FRACZ which does take place within this logic. FRACZ allows the effective formation of fractional values to occur where the arithmetic is otherwise integer only; FRACZ operates directly on BMET, whereas SMET includes the fraction effects indirectly through C1,2S.

Flow of function through the logic of Figure 3-9 proceeds as follows. The Correction, WB PLA is a regular logic array which forms three different correction pair outputs and the

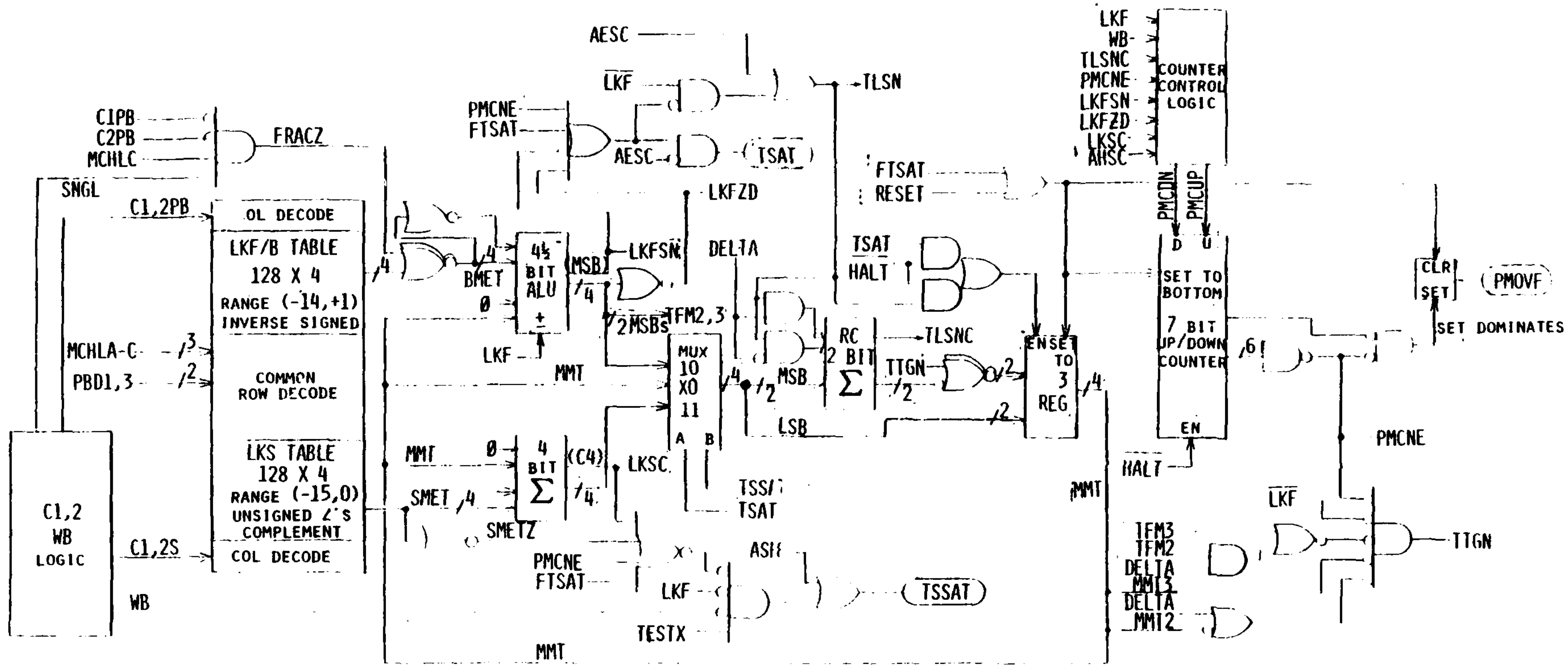
Figure 3-9. Correction, Worst Branch, Metric Logic Block Diagram



indicator output WB. The C1,2PB pair always reflect present branch decisions: these are new decisions in the case of "look forward" and previous decisions in the case of "look backward". The tentative decisions C1,2 always reflect the "next guess" of the decoder whether looking forward or backward (backward is in fact sideways with respect to the C1,2 corrections). The C1,2S corrections concern only the look sideways case (which is always concurrent with look backward). C1,2S represent the "next best guess" for a given branch and additionally encode fraction information for the SMET ROM. The correction PLA forms these outputs by considering incoming data (PBD1), search state (LKF), branch type (SNGL), mode (HARD), syndrome decoder state (S), and previous corrections (C1,2LST). The additional inputs TESTX and PBD0,2 are used for test mode only. The input PTAIL is used in conjunction with the other inputs only to form WB.

3.9 Path Arithmetic Processor

The Path Arithmetic processor shown in Figure 3-10 is the origin of search algorithm decisions in sequential decoding. The (path) arithmetic processor is not used in encoding. The primary inputs to this processor are the BMET and SMET numbers, while the most important outputs are the TSAT and TSSAT variables. The TSAT result is used in determining all the non-I/O-determined state transitions in the algorithm/control state sequencer (Section 3.2), while the TSSAT result is used in look-backward decoding situations only. The third most significant output from this processor is PMOVF, the path metric overflow indicator. When PMOVF occurs, the arithmetic capacity of the processor has been exceeded and a type of escape sequence is necessitated for the decoder; this event is clearly a very low probability event by design.



11/7/80

Figure 3-10. Path Arithmetric Processor

The logic flow in Figure 3-10 proceeds from left to right. First, the BMET and SMET numbers are added to the current metric-minus-threshold (MMT) value; BMET may be either added or subracted from MMT. The result of the BMET operation, combined with PMCNE (path metric counter not empty: or a state of the arithmetic processor), FTSAT (force TSAT: originated in the synchronization logic) and AESC (address (collision) escape) determine TSAT in each relevant decoding cycle. The SMET operation, combined with WB (worst branch), LKF (in this case signalling look back) and PMCNE, determine TSSAT. Additionally, the 4-bit results of the BMET and SMET operations are available for selection to further processing and eventual storage as the next MMT value when the algorithm/control logic (under TSAT and TSSAT control) so decides. This selection of next/tentative MMT takes place in the 3:1 multiplexer selected by TSAT,TSSAT. The further processing which may occur is either threshold-tightening, threshold-loosening or straight-through storage. Tightening (performed only in look forward under special conditions as signalled by the occurance of TTGN) always results in a new MMT of value zero. Loosening (performed only in look back under special circumstances where tightening has previously occurred, and hence where MMT has been selected at the 3:1 mux) utilizes the 2-bit adder to add either +4 or +8 to MMT as indicated by DELTA (constant in a given mode). Straight-through storage is just the loading of either the BMET or SMET result into the MMT register. There is some special logic concerning loosening (as indicated by TLSN) and the AESC input variable: while AESC always causes a threshold-loosening (useful only in block mode), AESC-originated TLSN's allow the 2 least significant bits of arithmetic result to propagate through rather than become zero as usual for loosening.

The rightmost features of Figure 3-10 are the path metric (extension) counter, its control logic and the PMOVF, PMCNE and TTGN logic. The path metric counter (PMC) is a modulo-16 extension of the 4-bit MMT register. When the PMC has contents greater than zero, the PMCNE variable so indicates. The PMC control logic considers the variety of conditions under which the PMC must increment or decrement. the PMOVF logic is a latch of an overflow condition in the PMC. The TTGN logic is a detect of threshold-tightening conditions which occur regularly in decoding.

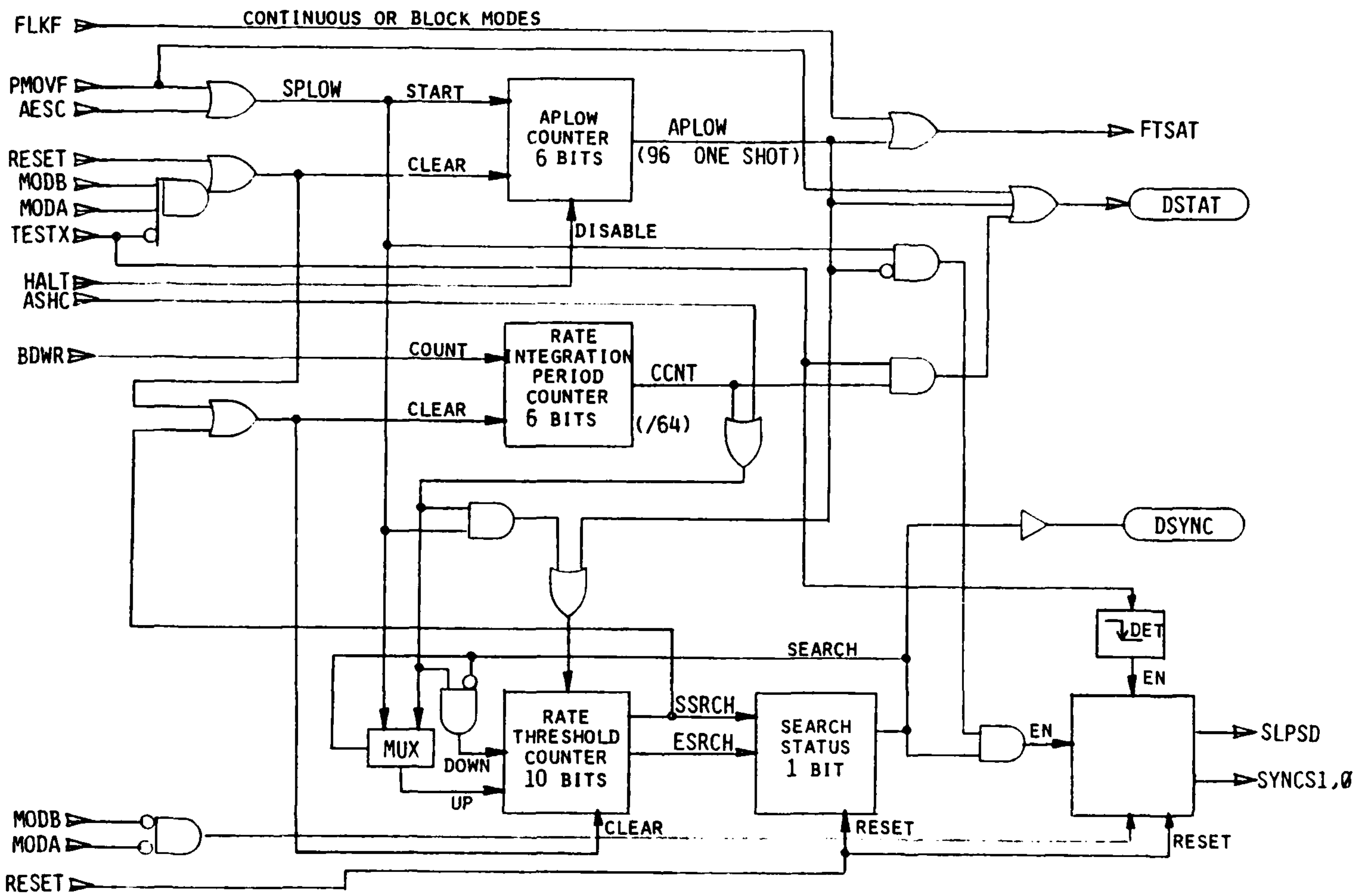
The entire arithmetic processor updates its state only on "active" decoding cycles as indicated by the false condition of HALT. The arithmetic processor resets its state whenever a (global) RESET or an (in-process) FTSAT occurs. In ordinary decoding, the state of this processor is updated only on the additional occurance of TSAT.

3.10 Synchronization Processor

The Synchronization (sync) processor determines the correct interpretation of incoming channel data in decoding. This function is necessary for continuous data decoding only and is unused in block operation or encoding. For reasons of convenience, the FTSAT signal originates in the sync processor and is used in block operation, but only as a reflection of FLKF which enters the sync processor as a general input. All other sync logic is reset continuously in block mode. The sync processor is block diagrammed in Figure 3-11.

The sync processor performs two functions: the generation of a "plow" (-forward) control signal APLOW and the detection of

Figure 3-11. Synchronization Processor



incorrect synchronization state followed by a change of sync state hypothesis. The inputs which may trigger the APLOW control are PMOVF and AESC. The APLOW control indirectly overrides decoding searches via the FTSAT output from this logic. The inputs which enter the detection of incorrect sync state are PMOVF and AESC, roughly measuring the difficulty of decoding a segment of received data, and BDWR, a direct measure of incoming data rate. The output controls SLPSD and SYNCS0,1 provide the means by which the sync processor revises the synchronization hypothesis.

The "plow" function of the sync logic is an automatic result of PMOVF or AESC, which represent arithmetic capacity failure or relative decoding progress (in the buffer memory) failure respectively. Either condition requires an immediate return to forward progress for nominally a code constraint length's worth of input branches: this is referred to as an auto-plow. The APLOW counter in Figure 3-11 acts as a digital one-shot of 56 cycles in duration.

The sync state revision function is the result of a long-term excess of decoding progress failures (AESC events) relative to incoming data inputs. The scaled data input rate is derived in the I/O weighting counter of Figure 3-11. This scaled rate is compared to the rate at which AESC occurs in the rate threshold counter of Figure 3-11. When the two rates have been integrated to a sufficient difference (threshold), the Search state is entered. In the Search state of the sync processor, each subsequent AESC event causes a revision of the sync hypothesis to occur. The overall sequence of hypotheses reflect the possible sync states of the particular mode in use (BPSK, QPSK, OQPSK, and

code rate are considered). When no AESC event occurs in 512 consecutive branch data input cycles during the Search state, the sync hypothesis in effect is declared valid and Search state is exited. Decoding then continues normally.

A subtlety exists in the use of the rate threshold counter of Figure 3-11. This counter actually performs in two unrelated modes. When not in Search state, this 10 bit counter is a simple up/down counter with up-direction overflow being the Search status toggle control. When in Search state, the upper 3 bits of this counter act to extend the 6 bit I/O weighting counter to 9 bits for the AESC-free timeout to exiting of Search state.

The sync state counter of Figure 3-11 and its associated SLPSD generator act to define and revise the sync state hypothesis. The sync state, SYNCSO, determines the interpretation of incoming branch data in the continuous data modes; it's effect is nullified in the I/O logic in block mode. Two possible I/O-interpretation sync states exist in each of the continuous modes, BPSK, QPSK, and OQPSK. The slip syndrome decoder control, SLPSD, handles ambiguity due to code branch type in code rates 3/4 and 7/8. SLPSD is continuously disabled in block modes. SLPSD allows single-branch "slips" of the syndrome decoder relative to the branch ID counter to accomplish revision of this timing hypothesis. There are 4 timing states in code rate 7/8, while only 2 in code rate 3/4.

There are 2 LS56 pin outputs originating in the sync processor: DSTAT and DSYNC. Each of these status outputs is used both in chip test mode and normal operation. DSTAT indicates path arithmetic overflows in block decoding and auto-plows in

continuous decoding. DSYNC indicates Search state decoding which is a warning status indicator in continuous decoding.

4. External Buffer Memory Timing and Configurations

4.1 Introduction

The LS56 decoder operates with an external 8-bit X N-location memory for the purpose of storing channel data and corrections. The number of locations in the memory varies depending on the operating mode, block or continuous, and the state of the variable, MBUFL. In block mode, the memory size is either 128 or 256 locations, and in continuous mode, the memory size is either 1024 or 4096 locations. The following is a detailed description of the memory timing and the different possible memory configurations of the LS56.

4.2 Memory Organization

During normal decoding, the LS56 has the possibility of doing a read and write to it's external buffer memory in a single decoder cycle. The basic timing for this is a read during the first half-clock cycle and a write during the second half-clock cycle.

The eight-bit buffer memory is partitioned into two four-bit nibbles. Bits 0-3 store the branch-data and are only written in I/O cycles. Bits 4-7 hold the decoder corrections in bits 6 and 7 and branch data in bits 4 and 5. This half of memory is written during any look-forward cycle to store new corrections and during any I/O cycle to store new branch data. The branch data is restored to memory during a look-forward cycle that is not an I/O cycle. The write control variables from the LS56 are WRAMA for bits 0-3 and WRAMB for bits 4-7. These are positive-sense write enables that are NAND-gated externally with CKPHB to generate the write enables for RAM.

4.2.1 Separate I/O RAM Interface

The LS56 is naturally set up for interface to separate I/O RAM. The RAM address outputs are connected directly to the RAM address inputs, RAMDI0-7 is connected to the RAM data outputs, and RAMDO0-7 is connected to the RAM data inputs. The write enables for the RAM are generated by gating WRAMA and WRAMB with CKPHB using a NAND-gate. The result is input to the WE/ input of the "A" and "B" halves of RAM. See Figure 4-1 for a logic diagram of the separate I/O RAM interconnect. The minimum timing for the RAM interface is shown in Figure 4-2.

The memory device types envisioned for use in this type of application include the Fairchild 93L422 for 256 or 128 X 8 operation and the Intel 2147 for 4094 or 1024 X 8 operation. Similar devices with adequate timing caracteristics may be substituted for these memory types.

4.2.1.1 Common I/O RAM Interconnect

Common I/O RAM would be used in applications where RAM timing is not critical so that the added time necessary to tri-state the LS56 RAM write bus will not affect the ability to access the memory. The RAM address out of the LS56, and the write-enable control for the RAM are connected in the same mannor as in the separate I/O RAM case. The common I/O data bus for the RAM is connected to the RAMDI inputs and the outputs of a set of tri-state drivers whose inputs are the LS56 RAMDO outputs. The output enable control for the tri-state drivers is the write-enable for the RAM. See Figure 4-3 for a detailed logic diagram of the common I/O memory interconnect. Figure 4-2 shows the RAM timing required for operation.

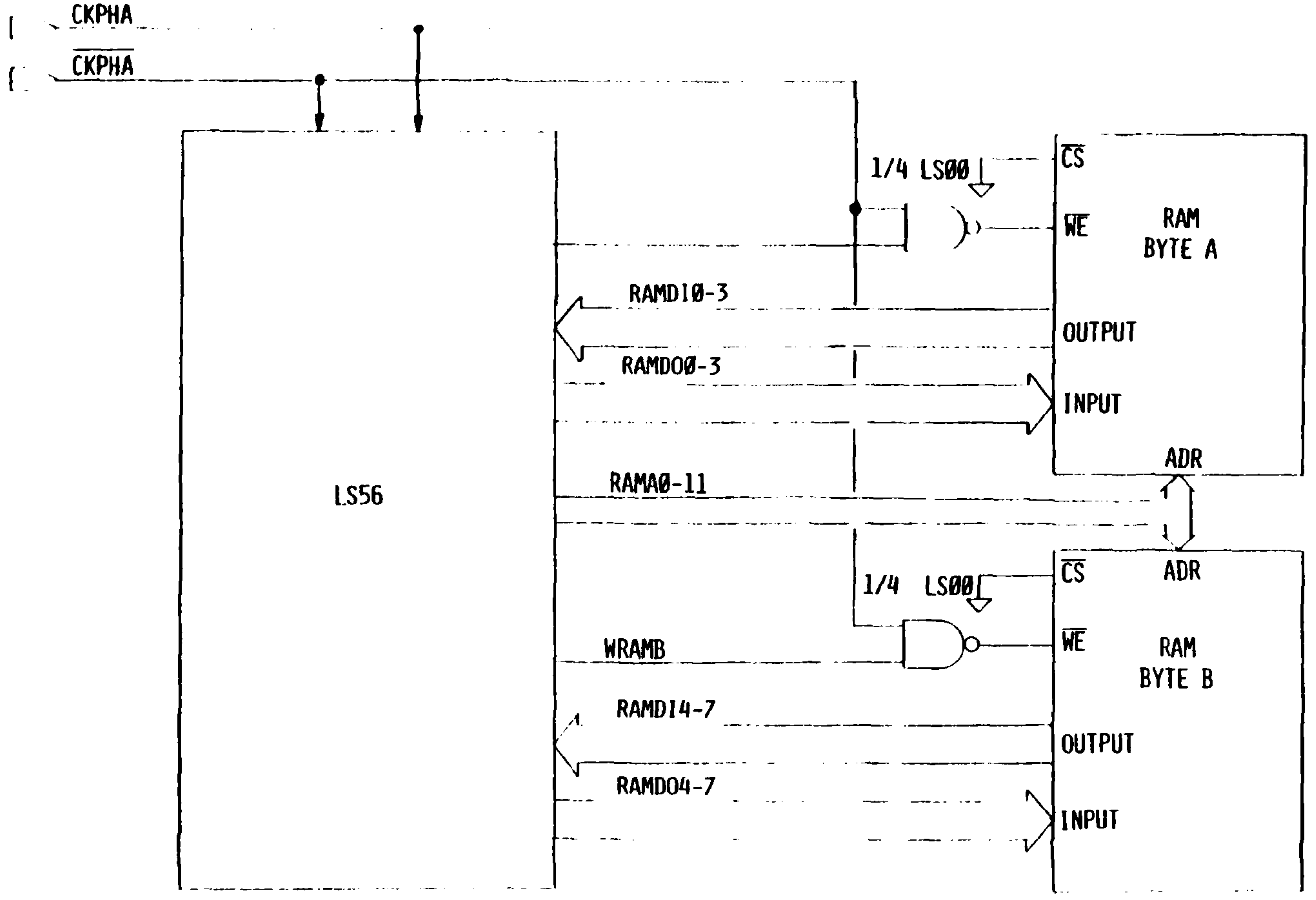
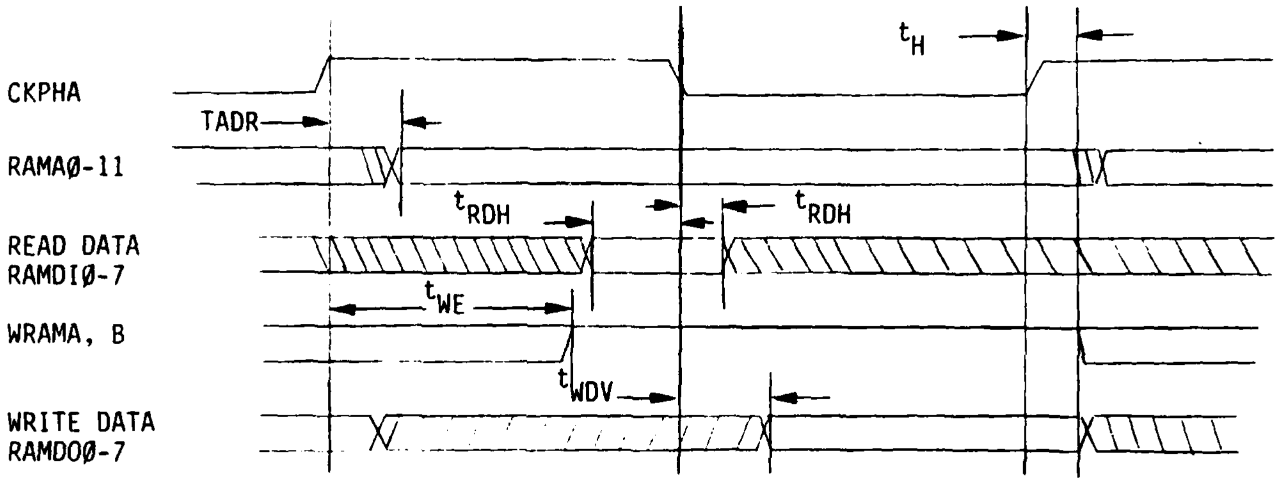


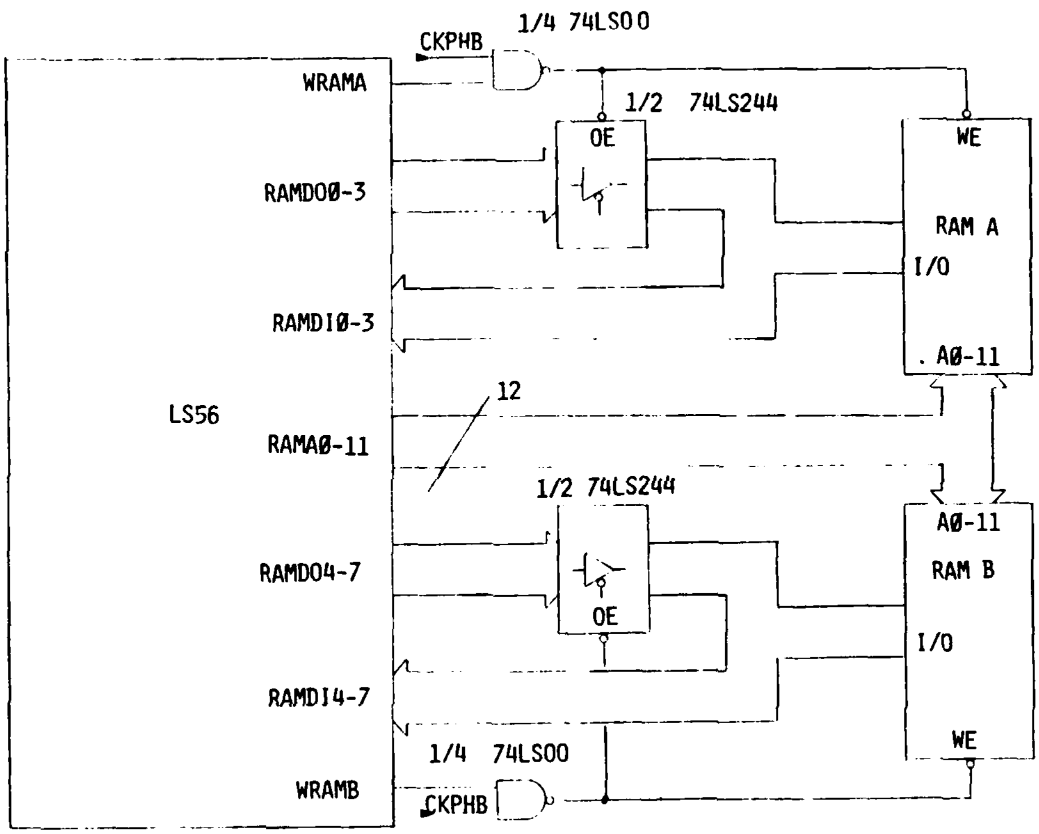
Figure 4-1. Separate I/O RAM Interconnect

Figure 4-2. RAM Timing



SYMBOL	DESCRIPTION	TIMING
t_{ADR}	ADDRESS VALID FROM RISING EDGE OF CKPHA	40ns MAX
t_{RDV}	READ DATA (INPUT) VALID PRIOR TO FALLING EDGE OF CKPHA	40ns MIN
t_{RDH}	READ DATA HOLD TIME AFTER FALLING EDGE OF CKPHA	10ns MIN
t_{WE}	WRAMA, B VALID AFTER RISING EDGE OF CKPHA	150ns MAX
t_{WDV}	WRITE DATA VALID AFTER FALLING EDGE OF CKPHA	40ns MAX
t_H	HOLD TIME ON ALL LS56 OUTPUTS	20ns MIN

Figure 4-3. Common I/O RAM Logic Interconnect



4.3 Alternative RAM Configurations

4.3.0.1 Use of One-Bit Wide Memories

In applications where one-bit memory devices are to be used, the write control for bits 4 and 5 of RAM may be controlled by WRAMA. This is because data written into bits 4 and 5 in both block and continuous mode is branch-data related and need only be written during I/O cycles. With bits 4 and 5 write enabled with WRAMB control, these bits are read and refreshed to memory during a non-I/O cycle. This is the critical delay path in the buffer memory interface. Removing the need for a read-refresh to RAM bits 4 and 5 allows the RAM interface to operate at higher computation rates.

4.3.0.2 QPSK Operation

Continuous mode QPSK operation requires only a six-bit wide buffer memory. The decoder can determine the sync-state of the demodulator with the information stored in bits 0-3 in RAM. Bits 6 and 7 are used for storing corrections, and bits 4 and 5 are not used. In the case of using 1-bit wide memory devices, this will reduce the device count.

5. Representitive Decoder Systems

To be written.

6. INTEGRATED SIGNAL CHARACTERISTICS

6.1 Type A Signal (Input)

The LS56 Type A input signal characteristics shall be compatible with external logic of the TTL-LS family. Specifically, the Type A input shall possess a max logic LOW voltage threshold of 0.8V and a min logic HIGH voltage threshold of 2.0V. The Type A input shall present a maximum DC load of +100 microamps in the HIGH state and shall not source more than -100 microamps in the LOW state. The input capacitance of the Type A input shall not exceed 25 pF.

The Type A input shall be typically driven by 74LS04 inverters.

6.2 Type B Signal (Output)

The LS56 Type B output signal characteristics shall be compatible with external logic of the TTL-LS family. Specifically, the Type B output shall provide a max logic LOW voltage of 0.5V with a minimum external DC load of 0.8 mA. The Type B output shall provide a min logic HIGH voltage of 2.5V with a minimum external DC load of -40 microamps. This provides that a Type B output drive a minimum of two 74LS00-typical gate input.

6.3 Type C Signal (RAM Addresses and Data Inputs)

The LS56 Type C signal characteristics shall be compatible with a worst-case buffer memory RAM configuration of eight parallel 93471-type RAM devices. The LS56 Type C output shall drive eight 93471 data or address inputs simulatneously. This requires that a Type C output provide a max logic LOW voltage of 0.5V when sinking a maximum DC load of 3.2 mA. The Type C output

shall provide a logic HIGH voltage between 2.4V and 4.5V limits when sourcing a maximum DC load of -320 microamps. Type C outputs above 4.5V shall incur increased DC loading up to a maximum of -8.0 mA due to 93471 RAM characteristics. Type C outputs shall operate with a maximum capacitive loading of 70 pF.

7. Parameters

7.1 Codes

The codes used in the LS56 have been independently selected for each code rate. Each code is specified in terms of two code generators, G1 and G2. Each code is systematic, and therefore G1 for all LS56 codes is the identity matrix. Each code is of code rate n-1/n (1/2, 3/4 or 7/8). The code rate 1 case is the trivial non-use of the syndrome decoder/encoder.

7.1.1 Code Generators

The G2 matrix specifies each code for its constraint length K. The values of K are always divisible by n-1. The G2 matrix represents the formation of parity bits from a canonical serial-shift-register encoder (see section 3.6). The G2 matrix is listed as G2(X,Y) where X refers to each n-1 bit segment of information data referenced to an arbitrary first segment of X=1. The Y dimension refers to the ordering of information bits within each segment with Y=1 specifying the first, or earliest, information bit input to the encoder. Thus, the G2 matrix can be "read" as a serial shift register connection specification (1=connection,0=no connection) in n-1 bit segments from first to last segment as X=1 to 13,21,36 in which each segment connection is naturally oriented for first (or input) encoder bit at left and last encoder bit as right.

The G2 matrices are given in Table 7-1.

The G2 matrices are implicit in the implementation logic of the syndrome decoder.

RATE = 7/8

X	G2(X,7)	G2(X,6)	G2(X,5)	G2(X,4)	G2(X,3)	G2(X,2)	G2(X,1)
X = 1	1	1	1	1	1	1	1
2	1	0	1	1	0	0	1
3	1	0	0	1	1	1	0
4	1	1	0	0	0	1	1
5	1	1	1	1	1	1	1
6	0	0	0	0	0	0	0
7	1	1	1	0	1	0	0
8	0	0	[illegible]	0	1	1	0
9	1	0	1	1	1	1	1
10	1	1	0	0	1	1	0
11	1	1	0	1	1	1	1
12	1	0	1	0	0	0	1
13	0	1	1	1	1	0	0
	S	D3		D2		D1	

RATE = 3/4

X	G2(X,3)	G2(X,2)	G2(X,1)
X = 1	1	1	1
2	0	1	1
3	1	1	0
4	1	1	1
5	1	0	1
6	0	0	0
7	1	0	0
8	0	1	0
9	1	1	0
10	1	1	1
11	0	0	1
12	1	1	1
13	0	0	0
X = 14	1	0	1
15	0	0	0
16	1	1	1
17	1	1	1
18	0	1	0
19	1	1	1
20	1	0	0
21	0	0	1
	S	D	

RATE = 1/2

X	G2(X)	X	G2(X)
1	1	X = 22	0
2	1	23	1
3	1	24	0
4	0	25	1
5	0	26	0
6	1	27	1
7	1	28	0
8	0	29	1
9	0	30	1
10	1	31	0
11	0	32	0
12	0	33	1
13	1	34	0
14	1	35	1
15	0	36	1
16	0		
17	0		
18	1		
19	1		
20	1		
21	0		

NOTE: All three codes are systematic in that the data is transmitted along with the parity so that the second generator matrix for each code is:

Table 7-1. G2 Matrices

7.1.2 Code Puncturing Format

The code rate 3/4 and 7/8 codes are "punctured" applications of a fictitious rate 1/2 code specified by the appropriate G2 matrix in Table 7-1. This technique is discussed in section 1 of this specification and is specified by the S and D underscores of the G2 columns in Table 7-1. The "S" columns are those positions in the encoder whose contents at the cycle of code-parity production (actually the latching of parity) are paired with the code-parity bit. The "D" column-pairs are those pairs of positions in the encoder whose contents are paired together as a "double" (hence "D") information branch. Each pair of encoder output bits is referenced to as a branch, regardless of code rate or modulation type. Regular sequences of "single" ("S") and "double" information branches constitute the encoded data. In rate 3/4, S and D branches alternate. In rate 7/8, the sequence is D1,D2,D3,S in that order and so on. Code rate 1/2 possesses only S branches of course.

The format of each encoded branch for all code rates, including the degenerate code rate 1 case, obeys the following. For S branches, information appears at DDATA and parity at DDATB in encoding. For D branches, the earlier information bit appears at DDATA, the later at DDATB in encoding. Encoded output from the LS56 always occurs in pairs. At the decoder input, this ordering is not in general preserved.

7.2 Arithmetic Quantities

7.2.1 Arithmetic Range and Overflow

The LS56 decoder evaluates search moves using additive arithmetic. Branch metric quantities (in binary) are added or subtracted from the path metric minus threshold (MMT) value of a decoding state. This MMT may have a range of [0,2047] with overflow occurring at 2047. These quantities and their limits are implicit in the implementation logic.

7.2.2 Thresholds

The arithmetic processor deals with the arithmetic state denoted MMT (see 7.2.1). The threshold implicit in the MMT binary number is of decimal value 8 for all decoding modes except rate 1/2 BSC (hard decision) and rate 7/8 soft decision, wherein it is 4.

7.2.3 Quantized Branch Metrics

The arithmetic processor adds and subtracts quantized branch metrics to the MMT (see 7.2.1) in binary. These quantized branch metrics are approximations of ideal real number metrics. All branch metrics fall in the range [-14,1] for all decoding modes. The method for implementing a specific set of branch metrics shall be on-chip ROM. The programming tables for the two ROM's (one "forward" and one "sideways") are shown in table 7-2. The interpretation of this table is the following: Branch-metrics are the forward metric values assigned to branches that have the indicated magnitude values and are to have the indicated corrections applied. These values are added to the path metric when the decoder moves forward and subtracted when the decoder moves backward from a given branch. Sideways metrics are the

values needed to allow the decoder to try the next-best correction hypothesis from the present hypothesis. The sideways metric is obtained from the values of old corrections applied to the branch and the magnitude bits associated with that branch. See section 3.8 for the block diagram of the correction and branch-metric logic.

Table 7-2. Compressed Branch Metrics

				RATE 7/8 SOFT MCHL=7		RATE 3/4 ERASURE MCHL=6	
CORRECTIONS 1	CORRECTIONS 2	MAGNITUDE BDATB	MAGNITUDE BDATD	BRANCH METRIC	SIDE METRIC	BRANCH METRIC	SIDE METRIC
0	0	1	1	1	-13	1	- 8
0	0	1	0	1	- 4	- 1	- 1
0	0	1	1	1	- 4	- 1	- 1
0	0	0	0	0	- 4	- 1	- 1
0	1	1	1	-13	0	- 8	0
0	1	1	0	- 4	- 9	- 8	- 8
0	1	0	1	-13	- 1	- 9	0
0	1	0	0	- 4	- 4	- 1	0
1	0	1	1	-13	- 1	- 8	- 6
1	0	1	0	-13	- 1	- 9	0
1	0	1	1	- 4	- 9	- 1	- 8
1	0	0	0	- 4	0	- 1	0
1	1	1	1	-14	-15	-14	-15
1	1	1	0	-14	-15	- 9	- 8
1	1	0	1	-14	-15	- 9	- 8
1	1	0	0	- 8	- 8	- 1	0

				RATE 3/4 HARD MCHL=5		RATE 3/4 SOFT MCHL=4	
CORRECTIONS 1	CORRECTIONS 2	MAGNITUDE BDATB	MAGNITUDE BDATD	BRANCH METRIC	SIDE METRIC	BRANCH METRIC	SIDE METRIC
0	0	1	1	1	- 7	1	0
0	0	1	0	X	X	1	- 4
0	0	1	1	X	X	1	- 4
0	0	1	0	X	X	0	- 4
0	1	1	1	- 7	0	-13	0
0	1	1	0	X	X	- 4	- 9
0	1	0	1	X	X	-13	0
0	1	0	0	X	X	- 4	- 4
1	0	1	1	- 7	- 7	-13	- 1
1	0	1	0	X	X	-13	0
1	0	1	1	X	X	- 4	- 9
1	0	0	0	X	X	- 4	0
1	1	1	1	-14	-15	-14	-15
1	1	1	0	X	X	-13	-14
1	1	0	1	X	X	-13	-14
1	1	0	0	X	X	- 8	- [illegible]

*Indicated sideways move not legal
(-3 is the recognition value for illegal move)

7-2 Continued

CORRECTIONS 1	CORRECTIONS 2	MAGNITUDE BDATB	MAGNITUDE BDATD	RATE 1/2 SOFT MCHL=3 BRANCH METRIC	RATE 1/2 SOFT MCHL=3 SIDE METRIC	RATE 1/2 ERASURE MCHL=2 BRANCH METRIC	RATE 1/2 ERASURE MCHL=2 SIDE METRIC
0	0	1	1	1	- 3*	1	- 3*
0	0	1	0	1	- 3*	0	- 3*
0	0	0	1	1	- 3*	0	- 3*
0	0	0	0	0	- 3*	- 1	- 3*
0	1	1	1	- 7	0	- 1	0
0	1	1	0	- 2	- 5	0	- 7
0	1	0	1	- 7	- 3*	- 7	- 3*
0	1	0	0	- 2	- 3*	- 1	- 3*
1	1	0	1	- 7	- 3*	- 6	- 3*
1	0	1	0	- 7	- 3*	- 7	- 3*
1	0	0	1	- 2	- 5	0	- 3*
1	0	0	0	- 2	0	- 1	0
1	1	1	1	-14	-15	-13	-14
1	1	1	0	-10	-11	- 7	- 7
1	1	0	1	-10	-11	- 7	- 7
1	1	0	0	- 5	- 5	- 1	0

CORRECTIONS 1	CORRECTIONS 2	MAGNITUDE BDATB	MAGNITUDE BDATD	RATE 1/2 HARD MCHL=1 BRANCH METRIC	RATE 1/2 HARD MCHL=1 SIDE METRIC	TEST MCHL=0 BRANCH METRIC	TEST MCHL=0 SIDE METRIC
0	0	1	1	1	- 3*	11	-12
0	0	1	0	X	X	- 3	- 4
0	0	0	1	X	X	- 7	- 8
0	0	0	0	-14	0	1	0
0	1	1	1	- 4	0	-13	-14
0	1	1	0	X	X	- 5	- 6
0	1	0	1	X	X	- 9	-10
0	1	0	0	-14	0	- 1	- 2
1	0	1	1	- 4	- 4	-12	-13
1	0	1	0	X	X	- 4	- 5
1	0	0	1	X	X	- 8	- 9
1	0	0	0	-14	0	0	- 1
1	1	1	1	- 9	-10	-14	-15
1	1	1	0	X	X	- 6	- 7
1	1	1	1	X	X	-10	-11
1	1	0	0	-14	0	- 2	- 3*

*Indicated sideways move not legal

8. Input/Output Protocols

8.1 Introduction

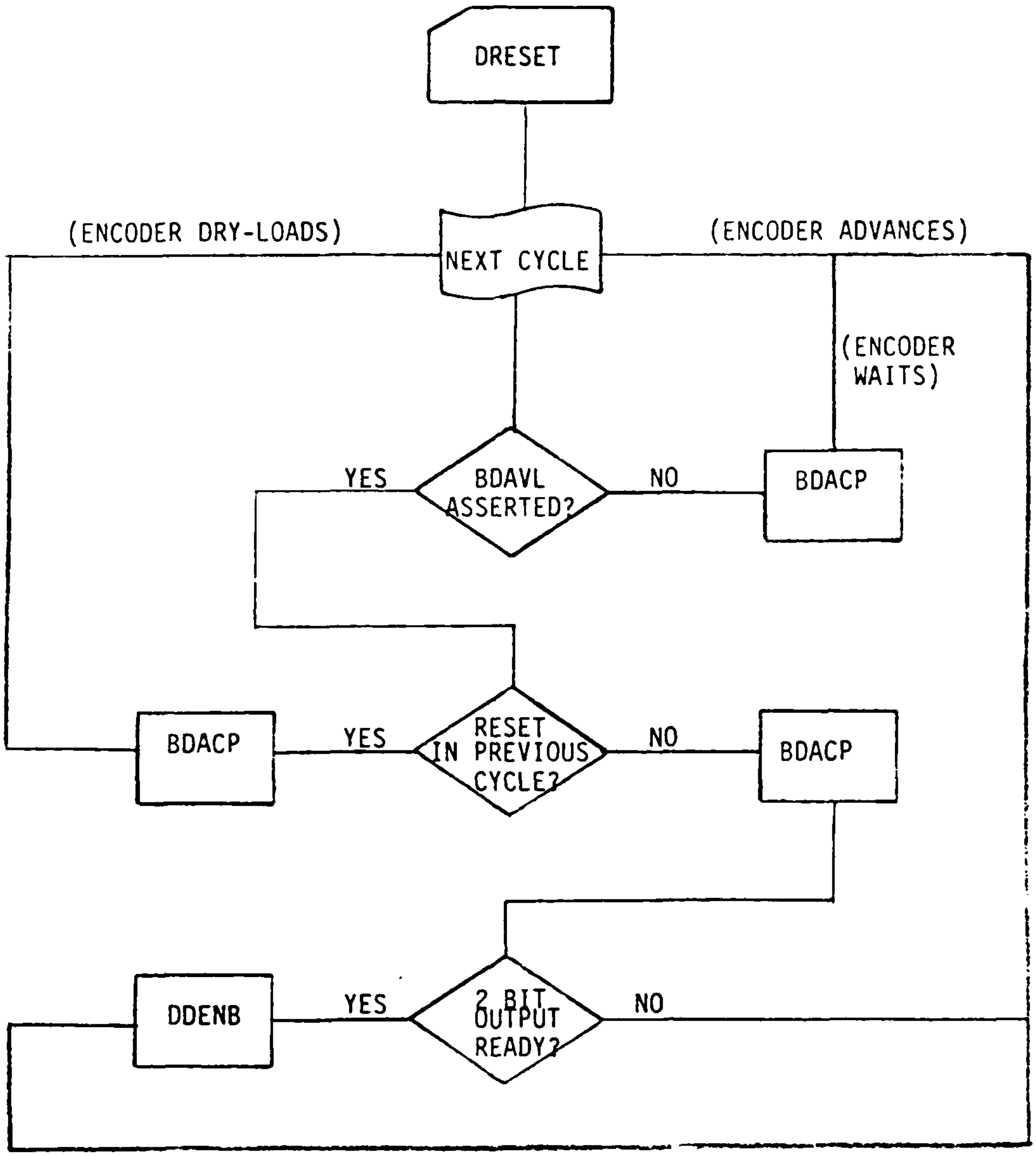
The following is a description of the input/output protocols for block, contiguous block, and continuous mode decoding and encoding.

8.2 Discrete Block Decoding Protocol

In discrete block encoding and decoding, the LS56 and associated hardware (e.g., DMA Interface IC, etc.) require start-of-block resetting, tail identification and, in the case of decoding, decoder buffer purging at the completion of block decoding. This decoding mode is the least efficient type possible (both structured TDMA and contiguous block operation are more efficient because the need for buffer-purging at decoding conclusion is eliminated). The corresponding encoding of discrete block is complicated only by tail identification since buffer memory is not utilized in encoding.

The encoding protocol for discrete blocks is presented in Figure 8-1. This flowchart indicates that an assertion of DRESET initializes the encoder state to all zeros and then proceeds with bit-by-bit information encoding as enabled by BDAVL. The data presentd at BDATA is encoded. All information inputs are acknowledged with a BDACP. All encoded symbol pairs output are indicated by DDENB; valid encoded output is only available as indicated by the assertion of DDENB. The external interface must identify input cycles (as indicated by BDAVL) as either ordinary data or block tail bits; this is done with external assertion of BTAIL for block tail inputs. It is unnecessary to apply ZERO data at BDATA in block tail, ZERO data is automatically assumed

Figure 8-1. Discrete Block Encoding Protocol



by the LS56. By identifying tail inputs with BTAIL, the LS56 is able to output only parity bits resulting from encoding; the known-ZERO tail data is not output. Hence, the resulting encoded block is naturally compact and contains no redundant information.

The decoding protocol for discrete blocks is presented in Figure 8-2. This flowchart indicates that an assertion of DRESET initializes the LS56 decoder to a state which corresponds to an initialized LS56 encoder. The LS56 decoder may also be automatically initialized at the conclusion of block decoding for the previous block, provided this previous decoding has been successful (i.e., not terminated). The entire decoding input/output protocol is enabled by BDAVL: this includes all inputting and outputting, whether associated with normal decoded data transfers or buffer-purging. The initial path of the LS56 is a return-to-next-cycle through the "decoder dry-loads" return branch. After this first unique path, the decoder either waits (BDAVL unasserted) or returns via the "decoder advances" branch. Eventually, the decoder will accept the last branch of received input data and the external interface will assert the DPURG signal. When this occurs, the PFLAG flag is set within the LS56 and search-free outputting occurs until the decoder returns to previously-decoded tail data. At this point, further outputting is disabled and the decoder rapidly reaches the buffer empty state. Note that BDAVL must continue to be asserted for this progression of events. When the empty buffer state is detected, the LS56 signals the external interface with a DDONE and automatically resets itself, ready for the next block of received data.

In decoding discrete blocks, LS56 algorithmic

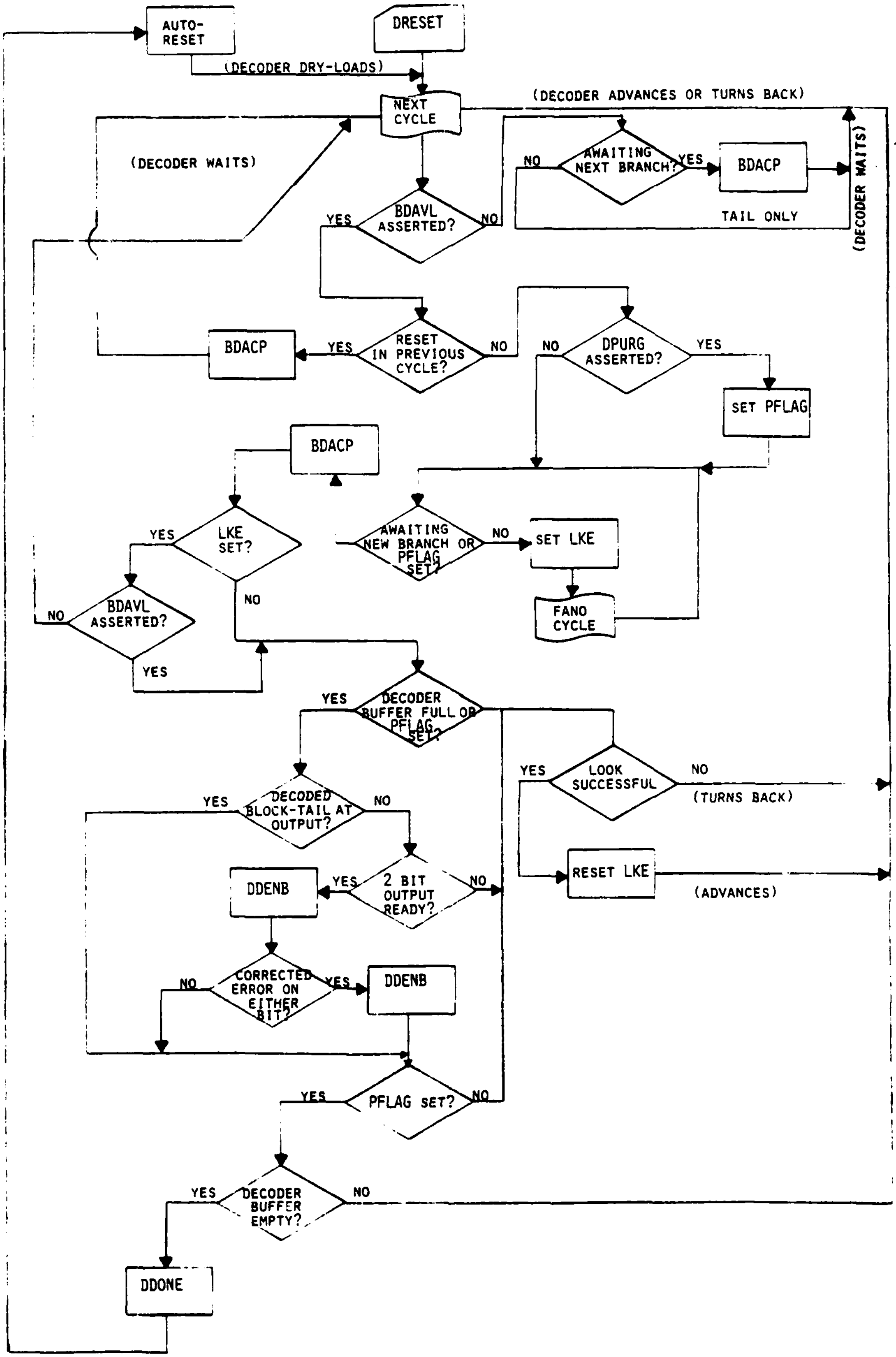


Figure 8-2. Discrete Block Decoding Protocol

characteristics require that one additional end branch pair of ZERO signs (BDATA,C) over and above the received tail parity branches be transferred from the external decoder interface. This requirement for an extra "pseudo branch" LS56 input guarantees block-terminal decoder behavior.

8.3 Contiguous Block Data Protocol

Contiguous block data operation is a close relative of discrete block data operation. In the contiguous case, two decoding features have been added which bear upon protocol and interface. Contiguous block data encoding is identical to discrete block data encoding with the exception that DRESETs need not accompany any but the first information block of a contiguous group of blocks.

The LS56 is able to economize its decoding overhead when individual received data blocks are available one after the other without appreciable delay between the output (from the LS56) of one block and the input (to the LS56) of the next block, i.e., contiguously. In this event, the external system may elect to utilize the contiguous block data protocol (potentially intercompatibly with discrete block transfers) to eliminate the need for buffer purging. To do this, the LS56 is designed to accept channel mode control (at MCHLA,B,C) as it would any other data (BDATA-D). When the inputting of one block ends, the external interface simply switches the MCHLA,B,C immediately following the LS56 acceptance of the last branch (pseudo-branch, see 11.2) of data. This allows contiguous blocks to bear independent coding rates and even mix hard and soft decision blocks.

In the canonical contiguous block decoder system (see 1.2.4), a pre-decoder channel buffer exists which may, with some small probability, reach incipient full condition as a result of a statistically extreme decoder advance rate (rate too small). This situation, imminent channel buffer failure of the block kind, requires the decoder system to override the LS56 and, essentially, force it forward into acceptance of new data. It may be necessary to force the LS56 forward the entire buffer length (128 or 256 branches) in order to obtain new branch data acceptance and hence relief from imminent buffer failure. It is noted that pratical decoder control buffer/interfaces must detect imminent channel buffer failure with at least 128 or 256 branches of latency to provide complete protection. Under these conditions the external interface may assert DPLOW to the LS56 which forces decoder advancing regardless of channel error correction. In fact, under the assertion of DPLOW, the LS56 corrects no errors and the resulting output data (essentially the next buffer length of output bits) will bear uncorrected channel errors. This is an unavoidable penalty and constitutes the primary source of system output error rate under contiguous block conditions.

The protocol for contiguous block decoding is shown in Figure 8-3. This flowchart is essentially identical to Figure 8-2 except for thedecision point "DPLOW asserted?". Here, the LS56 may bypass Fano search cycles, cycle by cycle, as long as DPLOW is asserted. The action of DPLOW is, then, a direct search control over the LS56 (i.e., realtime). Indication that DPLOW duration has been sufficient is gained by either a) counting out a buffer length of branches in DPLOW cycles or b) observing at least K (constraint length) input bit acceptances by the LS56.

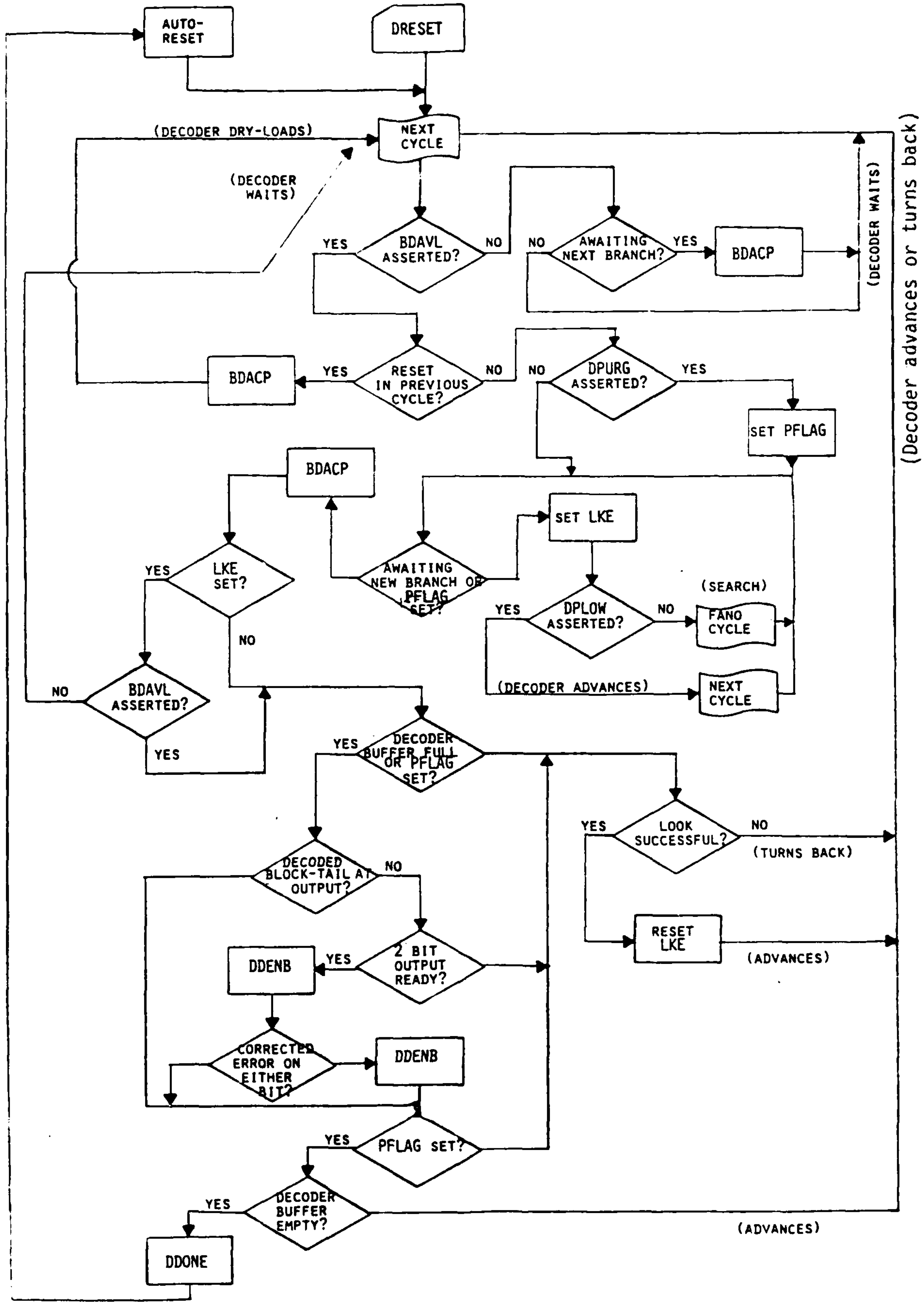


Figure 8-3. Contiguous Block Decoding Protocol

This translates to R**K/2 BDAVL/BDACP branch transfer cycles.

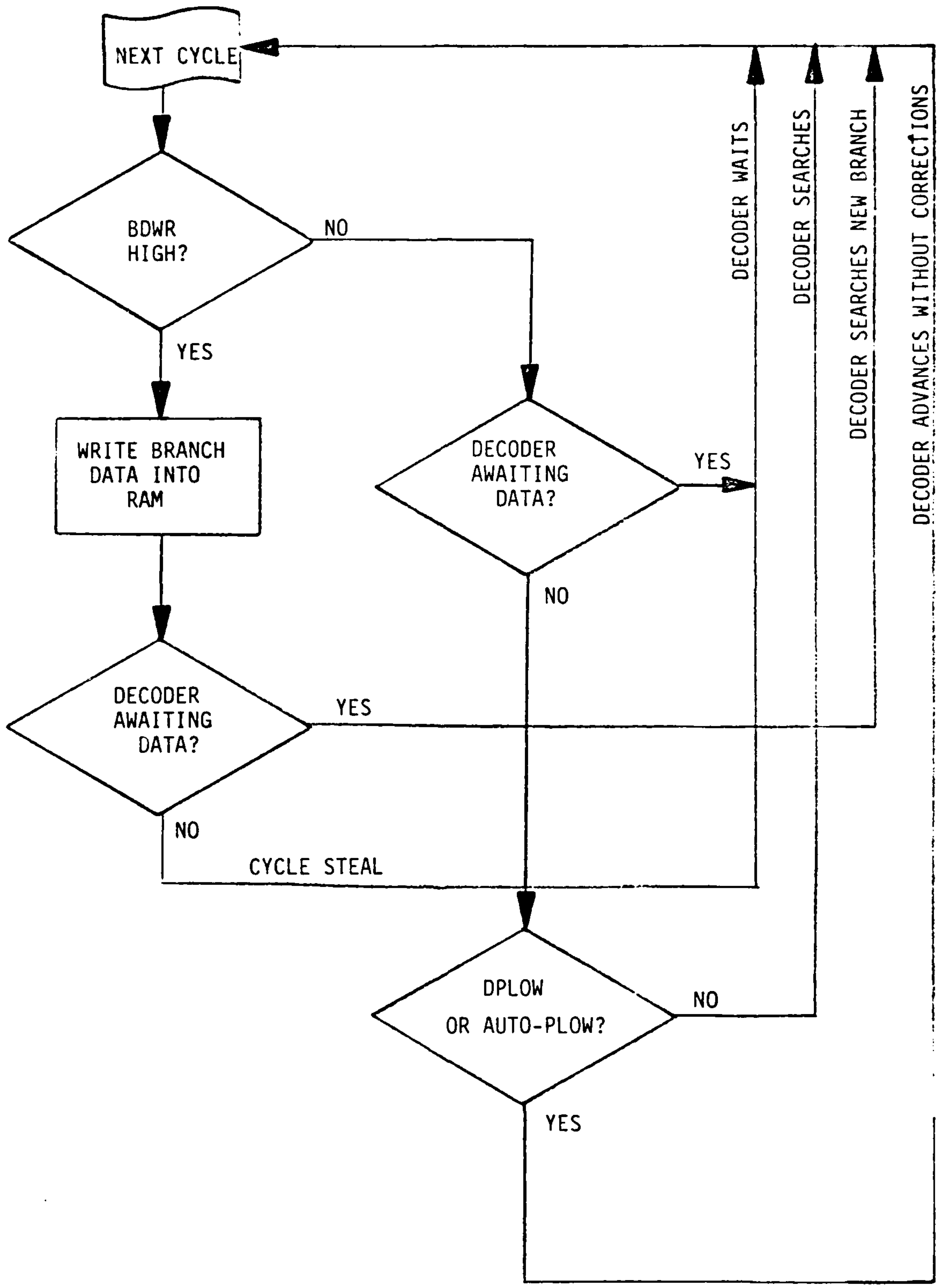
8.4 Continuous Mode Decoding

The contintous mode encoding and decoding protocols are very similar in implementation. Figure 8-4 shows a flow-idagram of the continuous mode I/O for encoding and decoding. Each are interrupt driven with a synchronous detect on the input data clock, BDAVL. The synchronous detect is accomplished by detecting the falling edge of the data input clock and the result is the occurance of the branch-data write variable, BDWR, in the cycle after the detect.

If an input is indicated by BDWR. the decoder must input the data and store it into the buffer memory during that cycle. Additionally, the data that was stored in RAM is read and output to the output buffer. During the input cycle, the deocder will search along a new branch of data if it was waiting for an input to move forward. If the decoder is on a search and not waiting for input, the deocder must hold it's present state and wait for the input cycle to end. Encoding I/O occurs in the same mannor as decoding I/O except the encoder is always waiting for data and will encode data during the I/O cycle.

If and input cycle is not indicated by BDWR, the decoder will wait for input if the decoder is caught up and needs input to move forward. The decoder will search if is is not caught up with the input process. The search cycle must take into account the plow-forward command that results from one of two sources: an internal auto-plow due to a buffer overflow, or from the external input, DPLOW. If a plow is indicated, the decoder moves forward without applying corrections to the data. With no plow

Figure 8-4. Continuous Mode I/O Protocol



indicated, the deocder executes a normal search cycle along one branch of data. The encoder will always wait for input if an input cycle is not indicated.

II. DMA/I Functional Specification

SH

DWG NO

APPLICATION	
NEXT ASSY	USED ON

REVISIONS			
REV	DESCRIPTION	DATE	APPROVED
		5/26/82	A S[illegible]

SHEET	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	44	48	49	50
REV																									
SHEET	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
REV																									

UNLESS OTHERWISE SPECIFIED DIMENSIONS ARE IN INCHES AND APPLY AFTER SURFACE TREATMENT

TOLERANCES
ANG ± 30'
.xx ± .03
.xxx ± .010

CONTR NO	
DWN	K. KUMM
QA	
ENGR	K. KUMM
RLSE	
DESIGN ACTIVITY APPROVAL	
DESIGN APPROVAL	

LINKABIT Corporation
10453 Roselle Street
San Diego, CA 92121

SPECIFICATION
DMA INTERFACE LSI
INTEGRATED CIRCUIT

SIZE	FSCM NO	DWG NO
A	11627	22134

SCALE	REV	SHEET	OF
	C		

LC336

Table of Contents

List of Figures

1. GENERAL

1.1 DMA Interface IC Objectives

The objective of the DMA Interface IC (hereinafter, DMA/I) is data-handling between a 16-bit microprocessor system in DMA mode and a custom LSI device known as the LS56. The LS56 has no microprocessor data bus interface capability; this is supplied by the DMA/I operating in conjunction with a DMA Controller IC, typically an Intel 8237, 8257 or a Motorola MC6844. See Figure 1-1 for the block diagram of a decoder system using the DMA/I with a DMA controller.

A number of specific functions are supplied by the DMA/I. These are:

1. Accepting a 16-bit DMA input word and "breaking it up" into the appropriate nibble-sizes required by the LS56.
2. "Feeding" these nibbles to the LS56 under control of a 2-signal handshake protocol: the so-called BDAVL/BDACP protocol.
3. Accepting 2-bit nibbles back from the LS56 from time to time.
4. Upon building up a complete 16-bit word using the LS56 output nibbles, inhibiting all LS56 input activity and executing a single word (16-bit) DMA output back to the data bus.
5. Outputing an 8-bit count onto the lower-byte of the data bus under control of a strobe signal, ECTEN/.
6. Coordinating DMA inputs and outputs with any of 3 DMA Controller IC types: Intel 8237, 8257 or MC6844.
7. Performing a small number of indirectly related tasks to save MSI ICs in the final system implementation.

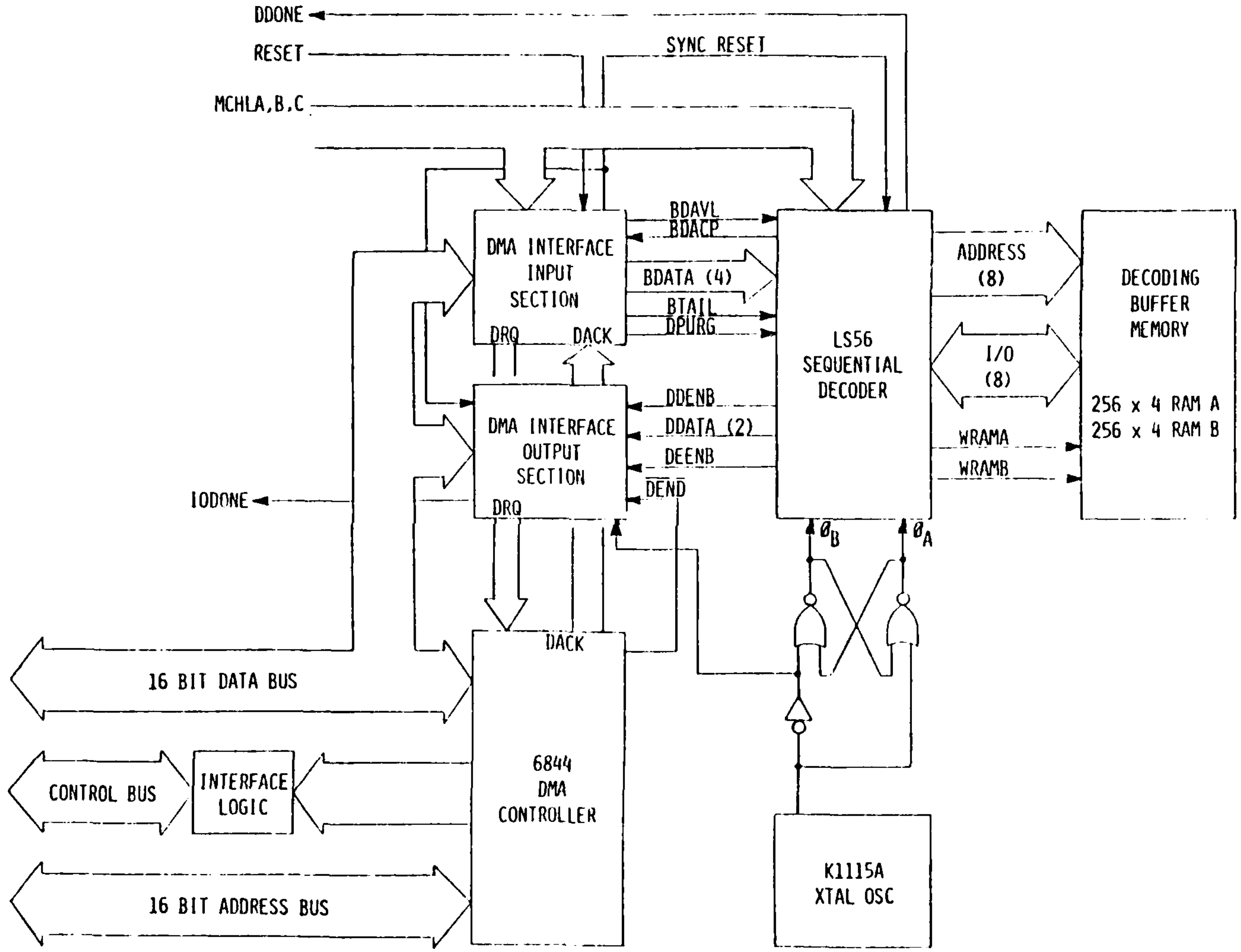


Figure 1.1. Packet Radio Sequential Decoder System Block Diagram

1.2 DMA/I System Environment

The DMA/I is nominal 64-pin IC (less test inputs and outputs) of Si-gate CMOS gate-array technology which interfaces to various other devices.

The DMA/I actually runs on 2 clocks. One clock supplies a majority of the circuitry, that portion of logic which handles DMA transfers; this clock is nominally 1.5 MHz in rate (1.6 MHz max).The other clock supplies the small indirect service-task logic mentioned in 1.1; this single-phase clock is the microprocessor system clock and will be nominally 8.0 MHz in rate (8.1 MHz max).

The DMA/I is intended to run in a single power supply environment. Only 5VDC ±5% power is available and all input signals will meet TTL logic threshold characteristics for the expected loading on each output.

The DMA/I and associated system will operate in an extended commercial-class environment. All device performance characteristics must be met over the -20°C to 75°C ambient temperature range. The device must be packagable in a hermetically-sealed ceramic carrier. Device reliability must be sufficient to allow eventual parts control program selection to MIL-STD-883B. Such selection procedures will not apply to the prototyping stage of DMA/I development.

2. INTERFACE SIGNALS

The following is a description of all the input and output signals for the DMA interface. See *Figure 2-1 for a block* diagram of the inputs and outputs.

2.1 Clock (CLKA)

The DMA/I shall receive one buffered clock for the purpose of clocking all flip-flops exclusive of the 8MHz portion of the logic. The minimum high logic level for each clock shall be 5.0V. The maximum low logic level shall be 0.2V. The DMA/I shall not load either CLKA or CLKB with a DC load greater than 40 mA nor an input capacitance greater than 150 pF. CLKA and CLKB will be of 50% $\pm$ 10% duty cycle. The maximum operating frequency shall be 1.6 MHz. All flip-flops are to be triggered on rising edges of CLKA. See Figure 2-2 for the logic to generate CLKA and CLKB.

2.2 Mode Control Inputs

2.2.1 Mode Channel A,B,C (MCHLA,B,C)

The DMA/I shall receive three buffered, positive-sense control signals which collectively define the rate and channel type of the decoder will operate on. MCHLA,B,C will be stable prior to the reset that initiates decoding a block of data, and will remain static until the block has been decoded.

The function of MCHLA,B,C is to provide information to the DMA interface about the type of data received from the processor and to be sent to the decoder on the BDATA-D port. Figure 2-3 shows the decodes of MCHLA,B,C and the data input word formatsfor

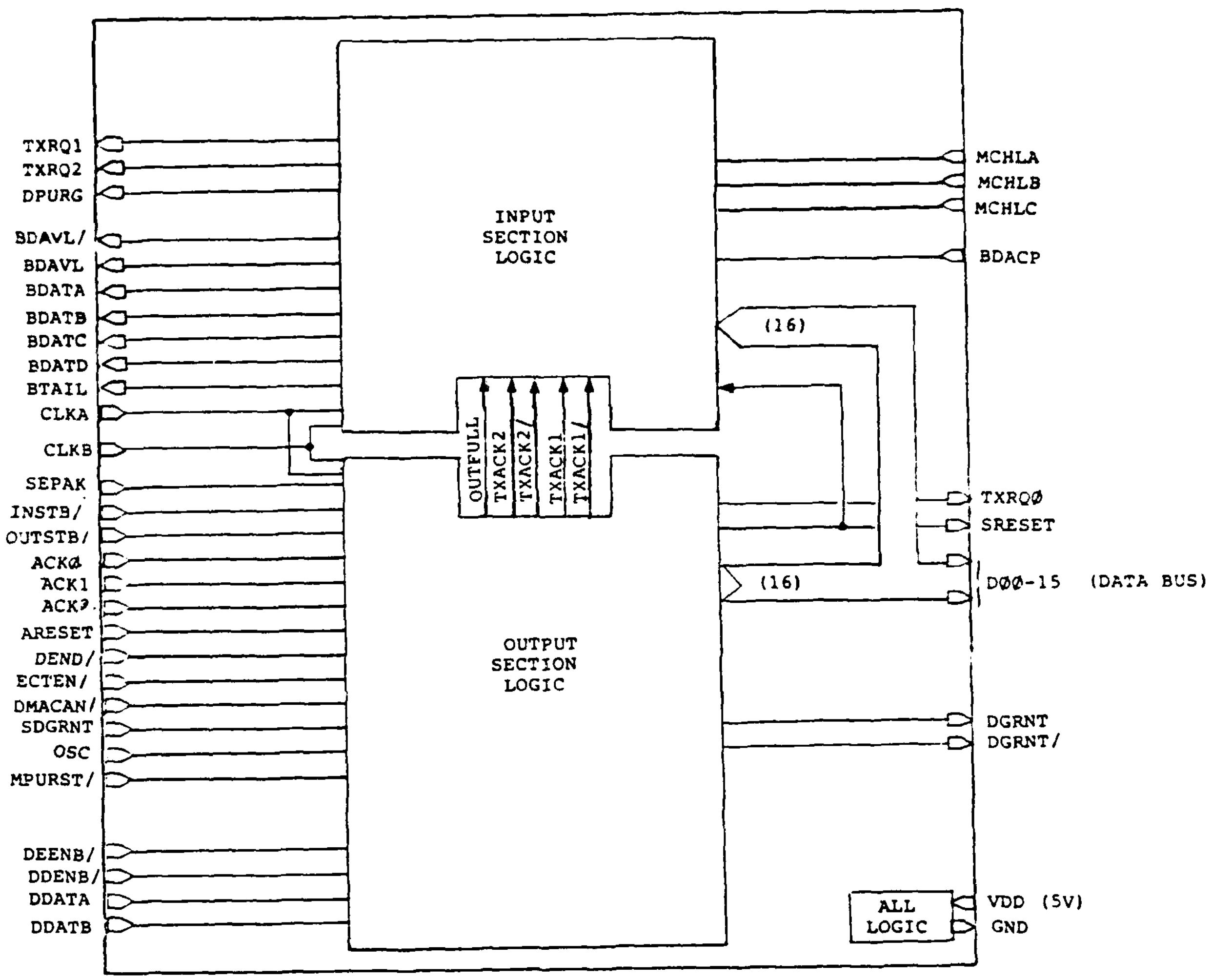


Figure 2.1. DMA Interface Input/Output

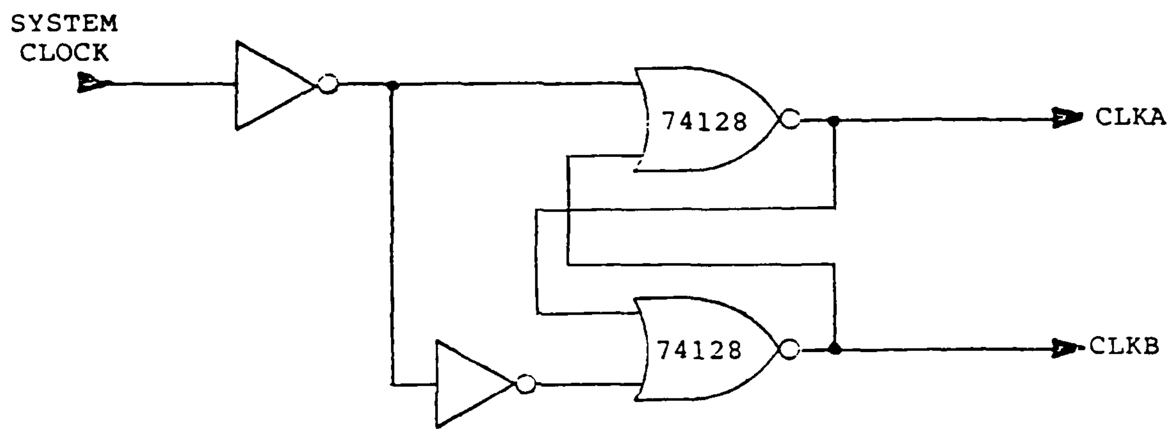


Figure 2.2. DMA Interface Clock Generation

each. See Figure 2-5 for a MCHLA-C timing. The timing diagram only indicates that MCHLA-C need be valid along with ARESET.

2.2.2 Separate/Decoded Transfer Acknowledge (SEPACK)

The DMA/I shall receive a buffered, positive-sense input signal, SEPACK, which defines the interpretation of the inputs, ACK0,1,2. SEPACK will be a static input during normal operation.

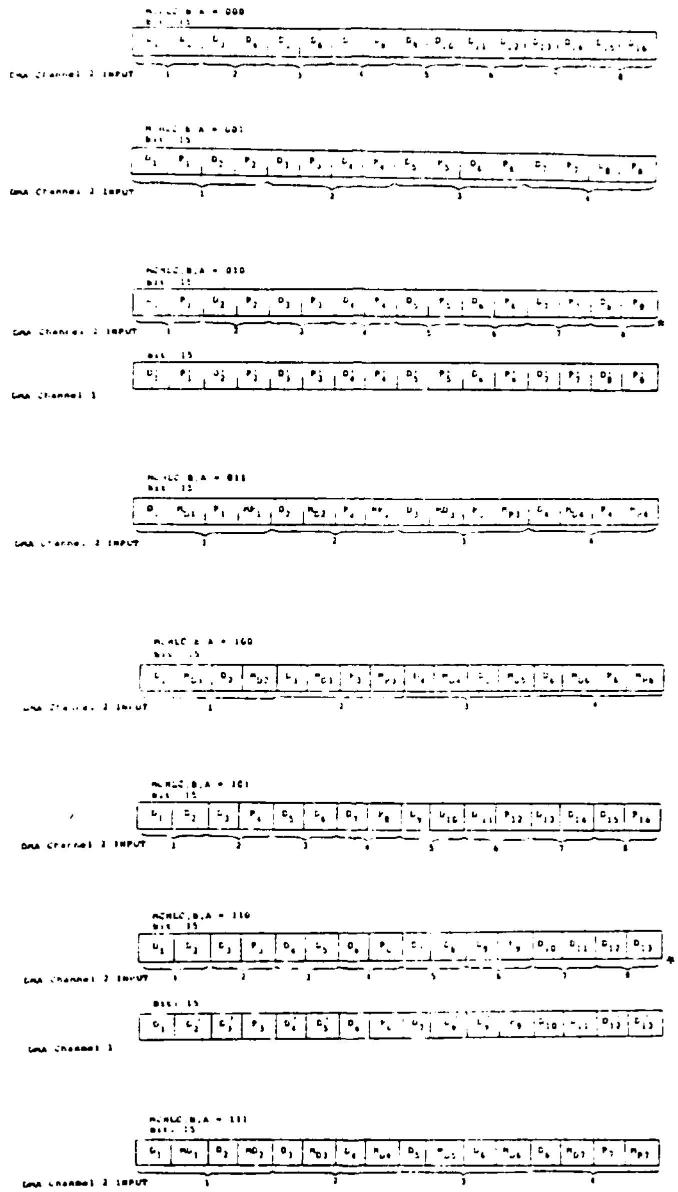
The function of SEPACK is to select between two possible transfer acknowledge protocols used on DMA controllers. With SEPACK low (logical 0), ACK0,1 are interpreted as a two-bit binary encoded enable which "steers" the strobe inputs, INSTB/ and OUTSTB/.

With SEPACK high (logical 1), ACKO,1,2 act as separate enables to steer the strobes INSTB/ and OUTSB/. For a more detailed description of the transfer acknowledge inputs, see Sections 2.5.2 and 2.5.3.

2.2.3 Asynchronous Reset (ARESET)

The DMA/I shall receive a buffered, positive-sense input, ARESET, which, when internally synchronized, resets all internal flip-flops that need resetting and generates the output SRESET. ARESET need not be internally captured by the synchronizing flip-flop if there is a minimum 100 ns set-up time prior to the rising edge of CLKA.

2.3 Decoder Interface Outputs



*See
Figure 2.4

Figure 2.3. DMA Input Word Formats

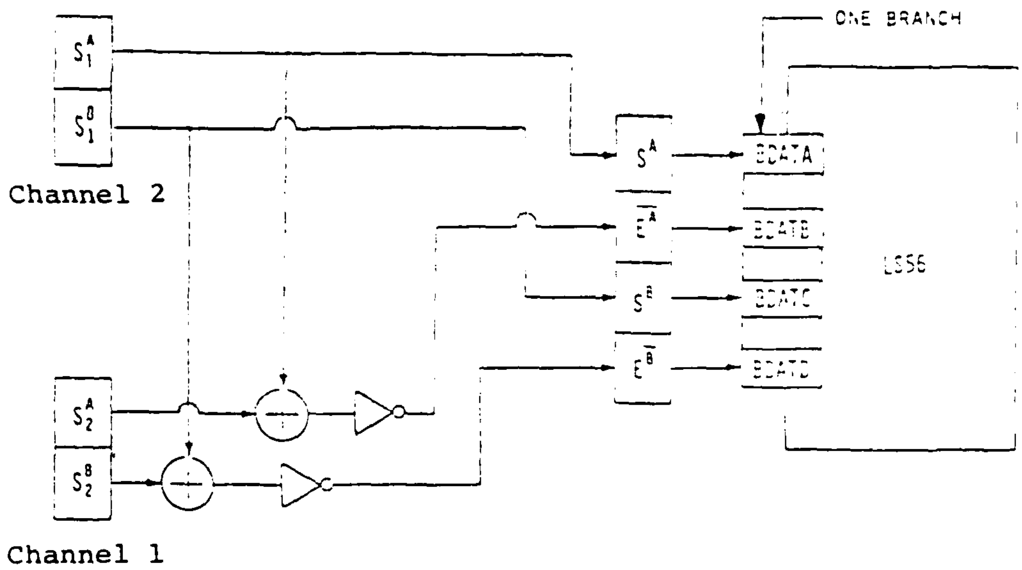


Figure 2.4. MCHLC,B,A = 010,110 Data Generation

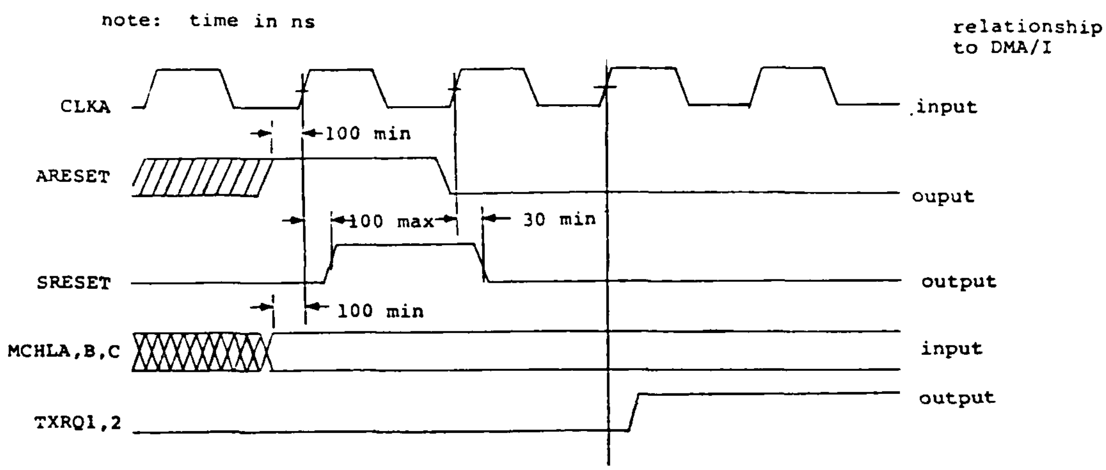


Figure 2.5. Reset, Mode Control, and Transfer Request Timing

2.3.1 Branch Data, BDATA,B,C,D

The DMA/I shall provide four buffered positive-sense data outputs to the decoder. BDATA,B,C,D shall be valid within 200 ns maximum from rising edge of clock and will hold 20 ns minimim after the rising edge of clock in any clock cycle for which BDAVL is active (see 2.3.3 below). BDATA-D shall be capable of driving 2 LS-TTL loads with 25 pf maximum capacitance.

2.3.2 Branch Tail (BTAIL)

The DMA/I shall provide a buffered, positive-sense output which, when at logic 1, identifies the current branch input data (BDATA-D) as belonging to the tail portion of the block. BTAIL is sychronous with the clock and shall be valid within 100 ns maximum from rising edge of clock and will hold 20 ns minimum after the rising edge of clock. BTAIL shall be capable of driving 2 LS-TTL loads with 25 pf maximum capacitance.

2.3.3 Branch Data Available (BDAVL, BDAVL/)

The DMA interface shall provide two buffered complimentary output signals indicating that the next branch of data on BDATA-D is ready to be transferred. BDAVL is synchronous in the clock and shall be valid 100 ns and will hold 20 ns after the rising edge of the clock, the BDAVL outputs shall be capable of driving 2 LS-TTL loads. BDAVL/ is the logical inverse of BDAVL.

The function of BDAVL is to provide a "data-ready" handshaking signal to the decoder. With BDAVL true, data is ready to be transferred. When the DMA interface receives BDACP true, the data transfer is complete and the DMA interface must either output another four-bits of data on BDATA-D if it can or reset BDAVL to its false state if the interface has no more data

to transfer. BDAVL also controls the output process from the decoder to the DMA interface, not allowing any data outputs from the decoder if BDAVL is false. See Figure 2-6 for a timing diagram of the decoder interface outputs.

2.3.4 Decoder Purge (DPURG)

The DMA interface shall provide a buffered, positive-sense output decoding-buffer purge control, DPURG. The DPURG control is synchronous and shall be valid 100 ns maximum after the rising edge of The clock. The DPURG output shall be capable of driving 2 LS-TTL loads with 25 pf maximum capacitance.

The function of DPURG is to cause the decoder to output the remaining data in its external buffer at the end of a block.

2.3.5 Synchronous Reset (SRESET)

The DMA interface shall provide a buffered positive-sense output, SRESET, that is a synchronized version of the input, ARESET. SRESET is the reset for all internal flip-flops that need resetting and the output provides the synchronous reset to the decoder. The act of SRESET going inactive initiates the first transfer request for data from the DMA controller. SRESET will be valid 100 ns maximum and hold 20 ns minimum after the rising edge of clock. SRESET shall be capable of driving 2 LS-TTL loads with 25pf maximum capacitance.

2.4 Decoder Interface Inputs

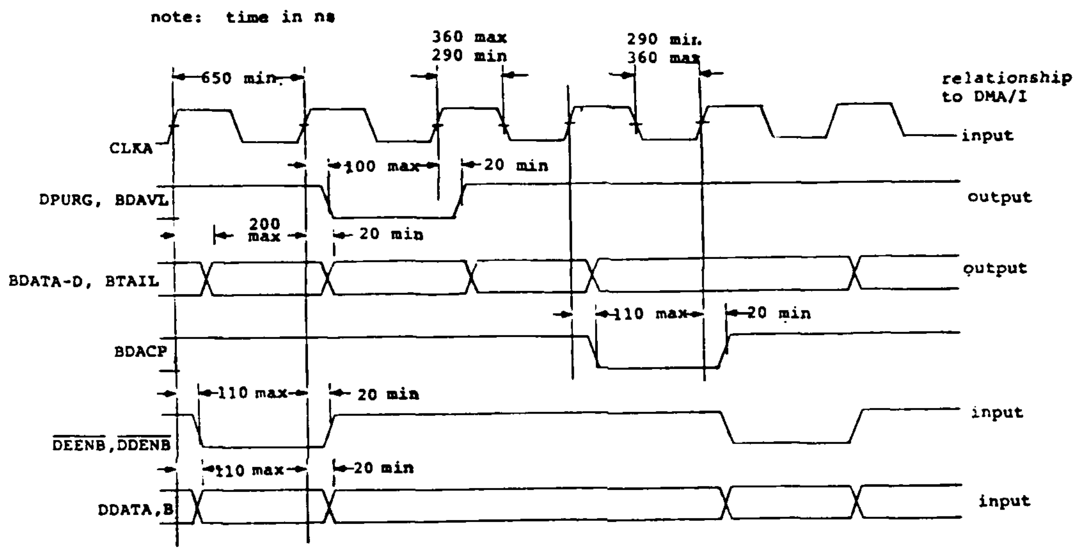


Figure 2.6. LS56 Interface and Control Timing

2.4.1 Branch Data Accept (BDACP)

The DMA interface shall receive a buffered, positive-sense input, BDACP, indicating that the decoder has accepted data if BDAVL is logical 1 or that the decoder is ready to accept data if BDAVL is logical 0. BDACP will be valid a maximum 100 ns and will hold a minimum 20 ns after the rising edge of the clock. See Figure 2-6 for the timing of the BDAVL-BDACP handshake.

2.4.2 Decoded Data (DDATA,B)

The DMA interface shall receive two buffered, positive-sense inputs, DDATA,B, that represent decoded data from the decoder. DDATA,B are synchronous inputs, valid 110 ns maximum and will hold for a minimum of 20 ns after the rising edge of the clock. The DDATA,B inputs are to be captured by the DMA interface any time the synchronous enable, DDENB/ is active.

2.4.3 Decoded Data Enable (DDENB/)

The DMA interface shall receive a buffered, negative-sense input, DDENB/, that identifies rising edges of the clock that data on DDATA,B is valid and latched in the DMA interface. DDENB/ is valid 110 ns maximum and will hold 20 ns minimum after the rising edge of clock. See Figure 2-6.

2.4.4 Decoded Error Enable (DEENB/)

The DMA interface shall receive a buffered, negative-sense input, DDENB/, that is a synchronous enable for the first stage of the error counter. DEENB/ is valid 110 ns maximum and will hold 20 ns minimum after the rising edge of the clock.

2.5 DMA Controller Interface

2.5.1 Transfer Request (TXRQ0,1,2)

The DMA interface shall provide three buffered, positive-sense outputs, TXRQO,1,2, to request a DMA transfer from the DMA controller. TXRQO,1,2 shall be valid no longer than 80 ns after the request is acknowledged from the DMA controller. TXRQO,1,2 shall be capable of driving 2 LS-TTL loads with 20pf maximum capacitance.

The transfer request outputs generate requests for transfer of data to and from the decoder system. TXRQO is the request for an output data transfer from the decoder system. TXRQ1 and TXRQ2 request data input to the decoder system. TXRQ1 is used in BEC mode and is disabled in BSC and soft decision mode. TXRQ2 is used in all operating modes. TXRQ1 and TXRQ2 are set active, depending on operating mode, the cycle after SRESET goes away, thus initiating data transfers to the decoder system. See Figure 2-5 for this timing.

2.5.2 Transfer Acknowledge (ACK0,1,2)

The DMA interface shall receive three buffered inputs, ACK0,1,2, that acknowledge the transfer request, depending on the state of SEPACK and the input/output strobes, INSTB/ and OUTSTB/. ACK0,1,2 will set-up 0 ns minimum and hold 100 ns minimum relative to the valid (logical 0) state of INSTB/ and OUTSTB/. See Figure 2-7 for the timing relationships.

With SEPACK a logical 0, the acknowledge inputs, ACK0,1, are a positive-sense two-bit binary encode of the channel to be serviced by the request. Under these conditions, ACK2 must be

note: time in ns

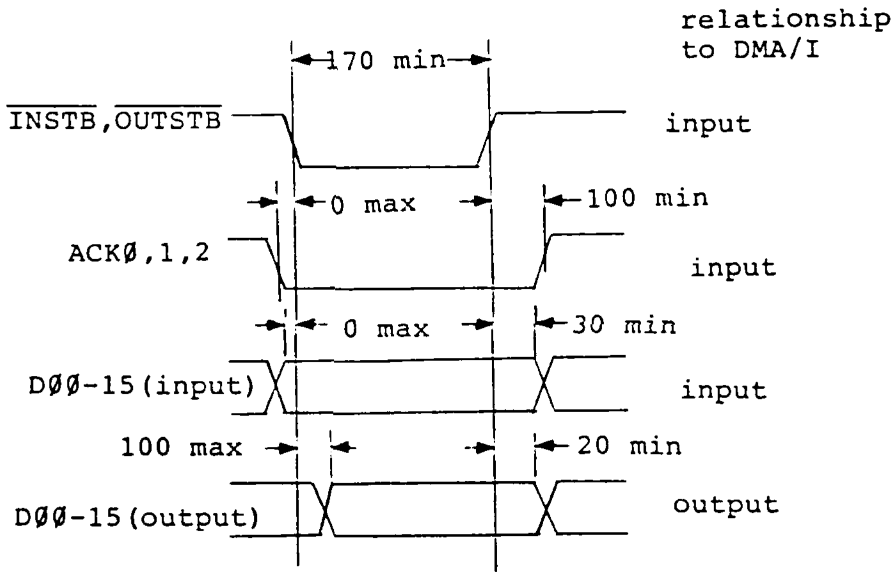


Figure 2.7. DMA Input/Output Timing

held at logical 0. The request is acknowledged when the binary encode is accompanied by the combination of INSTB/ and OUTSTB/ low simultaneously. This type of acknowledge is used with the Motorola MC6844 DMA controller.

When SEPACK is a logical 1, the acknowledge inputs, ACKO,1 2 are negative-sense seperate acknowledges for each channel. The request for channel 0 is acknowledged by ACK0 and OUTSTB/ low simultaneously. A request by channels 1 or 2 is acknowledged by ACK1 or ACK2 low along with INSTB/ low. This type of acknowledge is used with the INTEL 8257 or 8237 DMA controller.

2.5.3 Input/Output Strobes (INSTB/, OUTSTB/)

The DMA interface shall receive two buffered, negative-sense inputs, INSTB/ and OUTSTB/, for strobing data to and from the DMA controller. OUTSTB/ and INSTB/ will be a pulse with minimum width 170 ns during the time that one of the acknowledge inputs (ACK0,1,2) is active. See Figure 2-7 for the timing of INSTB and OUTSTB/ relative to data and the acknowledge inputs.

OUTSTB/ is the strobe from the DMA controller indicating an output transfer from the DMA interface is in process when it is in the logical 0 state. On the low-to-high transition of OUTSTB/, the transfer is complete. OUTSTB/ is used only with the acknowledge for channel 0.

INSTB/ is the strobe from the DMA controller indicating when it is in the active (low) state, that data is being transferred to DMA interface. On the low-to-high transition of INSTB/, the data is latched in the DMA interface and the transfer complete. INSTB/ is used with the acknowledge for channels 1 and 2.

The INSTB/ and OUTSTB/ inputs are tied together with SEPACK is low when interfacing to the Motorola 6844 DMA controller. The TXSTB/ output from the DMA controller is connected to both these inputs.

With SEPACK high, INSTB/ and OUTSTB/ are connected to the IOW/ and IOR/ outputs from the DMA controller respectively. The DMA controller used in this configuration is the INTEL 8257 or 8237.

2.5.4 DMA End (DEND/)

The DMA interface shall receive a buffered, negative-sense input, DEND/, to indicate the last transfer from DMA channel 0 is complete. The width of the DEND/ pulse shall be no less than 200 ns. When DEND/ is in the active (low) state, along with the last transfer being complete from channel 2, the DMA interface changes the control over the decoder to the next phase in decoding a block of data.

2.5.5 Error Count Enable (ECTEN/)

The DMA interface shall receive a buffered, negative-sense input, ECTEN/ for the purpose of strobing the error count from the 8-bit counter onto data bus, D00-D07. The minimum width for ECTEN/ is 170 ns. The error count must be valid on the data bus 100 ns maximum after the high-to-low transition of ECTEN/.

2.6 Processor Interface

2.6.1 Microprocessor Clock (OSC)

The DMA interface shall receive a buffered, positive-sense clock input, OSC, for the purpose of clocking the DGRNT flip-flop. OSC shall be a maximum 8.1 MHz and have a 50% + 12% duty cycle. See Figure 2-8 for OSC timing.

2.6.2 Set DMA Grant (SDGRNT)

The DMA interface shall receive a buffered, positive-sense input, SDGRNT, as the "set" input for the DGRNT flip-flop. SDGRNT shall have a minimum set-up time of 25 ns before the falling edge of OSC. SDGRNT is an input from the processor indicating that the DMA controller may take control over the data bus.

2.6.3 DMA Cancel (DMACAN/) and Processor Reset (MPURST/)

The DMA interface shall receive two buffered, negative-sense inputs, DMACAN/ and MPURST/, for the purpose of externally resetting the DGRNT flip-flop. The width of DMACAN/ and MPURST/ will be a minimum 3 OSC cycles in length.

2.6.4 DMA Bus Grant (DGRNT)

The DMA interface shall output two buffered, complimentary outputs, DGRNT and DGRNT/. For the purpose of indicating to the processor and the DMA controller that the DMA interface has control of the data bus. DGRNT and DGRNT/ shall be valid 80 ns maximum after the falling edge of the OSC clock. DGRNT and DGRNT/ shall be capable of driving 2 LS-TTL loads and drive a maximum capacitive loading of 20 pf.

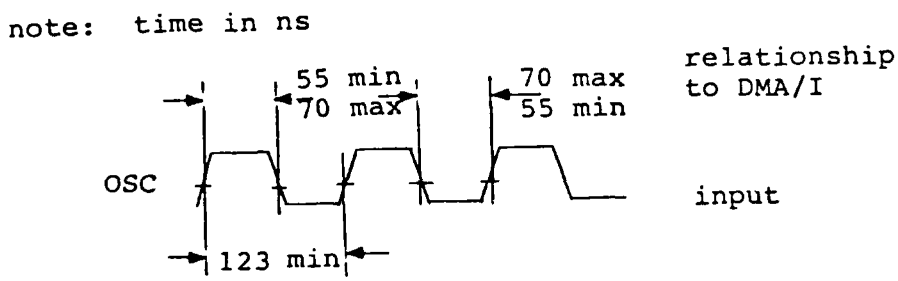


Figure 2.8. OSC Timing

2.6.5 Data Bus (D00-D15)

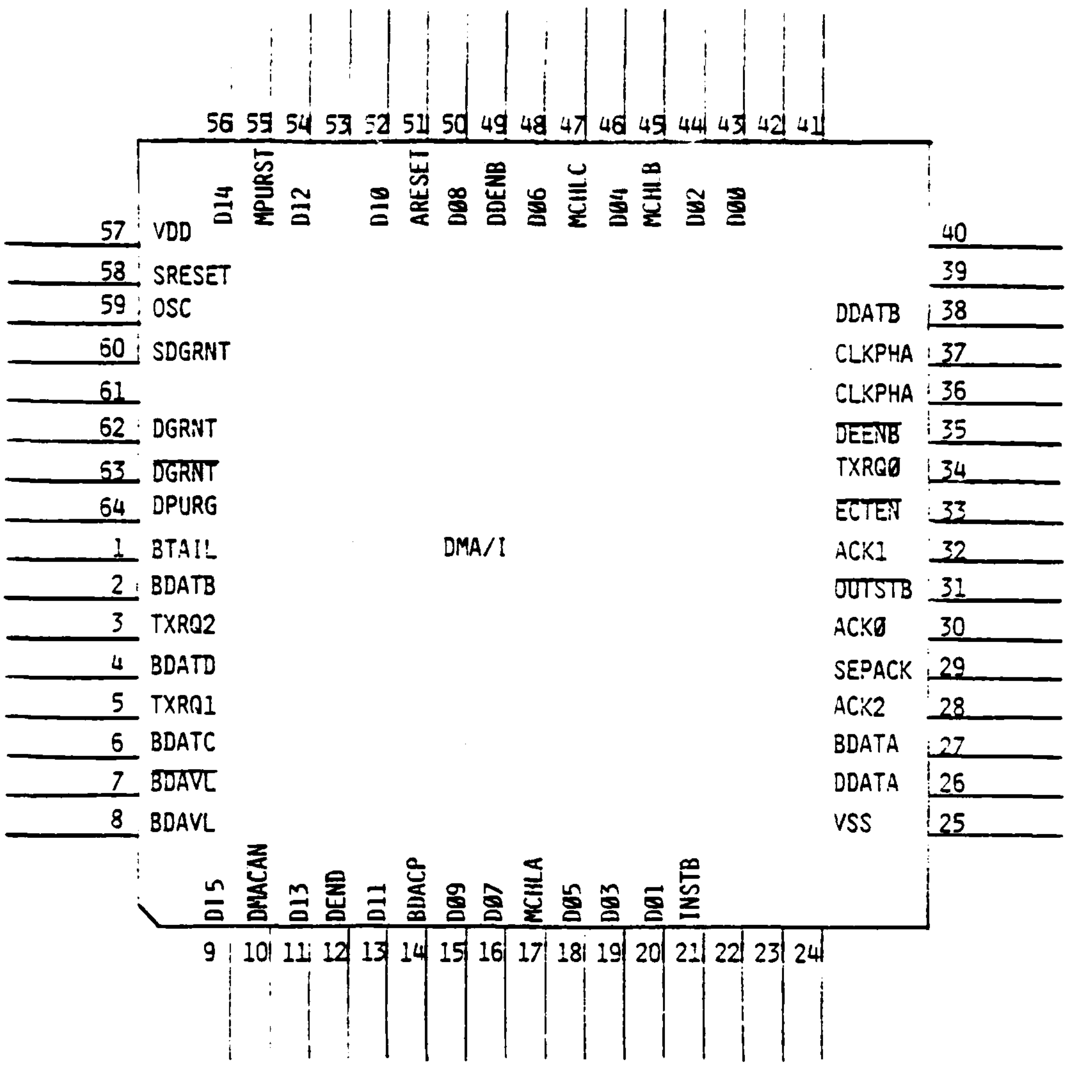
The DMA interface shall provide a 16-bit tri-state input/output data bus, D00-D15, for transmission of data between the DMA interface and the processor. Output data must be valid on the data bus 100 ns maximum after the high-to-low transition of ECTEN/ or, while acknowledging a channel 0 transfer, OUTSTB/. Data shall hold 20 ns minimum after a low-to-high transition of ECTEN/ or OUTSTB/. Input data will be valid 0 ns maximum after INSTB/ transits high-to-low. There will be 30 ns minimum hold time on the data for a low-to-high transition of INSTB/ while acknowledging a transfer on channels 1 and 2. See Figure 2-7 for the data input/output timing. The D00-D15 outputs shall be capable of driving 2 TTL inputs with a maximum capacitive loading of 50 pf.

3. Packaging Requirements

The prototype DMA/I is packaged in a 64-pin leadless chip carrier. The contact spacing for this carrier is 0.040" with 16 contacts on each side of the package. The pin-out description for the DMA/I in this package is shown in Figure 3-1.

Future production of the DMA/I will be packaged in a 68-pin leadless chip carrier with 0.050" spacing between contacts. This package shall meet JEDEC type A standards for LSI packages and is the same package used in the LS56.

Figure 3-1. DMA/I Logic Symbol



I.

I.1 D.C. Output Specification

Under Maximum Loading: V_{OL} = 0.4V max
V_{OH} = 2.4V min

Capacitive Loading on Outputs:

BDAVL, BDAVL, BDATA,B,C,D, BTAIL, SRESET, DPURG	C_L = 25pf max
TXRQ0,1,2, DGRNT, DGRNT	C_L = 20pf max
D00-D15	C_L = 50pf max

All outputs shall be capable of driving MOS inputs and meet the timing specifications.

Outputs except for D00-D15 shall be capable of driving 2 LS-TTL loads.

D00-D15 shall be capable of driving 2 standard TTL loads.

I.2 A.C. Timing Test Points

Inputs:	CLKA,B	V_{IL} = 0.5V V_{IH} = 4.5V
	INSTB, OUTSTB	V_{IL} = 0.8V V_{IH} = 2.0V
	All Other Inputs	V_{IL} = 0.4V V_{IH} = 2.8V

Outputs: V_{OL} = 0.6V
V_{OH} = 2.2V

I.3 Input Rise/Fall-Time Specifications

CLKA,B	$t_r = t_f$ = 40 ns (0V to 5V)
INSTB, OUBTB	$t_r = t_f$ = 40 ns
ECTEN, OSC	$t_r = t_f$ = 15 ns
SDGRNT	$t_r = t_f$ = 20 ns